VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Systém prevencie proti narušeniu
Intrusion Prevention System

2012 Bc. Michal Krupa

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Michal Krupa**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Systém prevence proti narušení
Intrusion Prevention System

Zásady pro vypracování:

Cílem diplomové práce je navrhnout a ověřit systém prevence proti narušení, tzv IPS. V práci je možno využít open source řešení založených na iptables a projektu SNORT v online režimu, nebo projektu bro-ids. Navržený systém bude testován pomocí různých penetračních nástrojů, např. hping, nmap apod.

1. Úvod do problematiky IDS/IPS systému.
2. Popis a vlastnosti firewallu využívajícího iptables.
3. Návrh vlastního řešení systému IPS.
4. Testování navrženého řešení pomocí penetračních nástrojů.

Seznam doporučené odborné literatury:

Rehman, R. *Intrusion Detection Systems with Snort*, Pearson Education 2003, ISBN 0-13-1400733-3
Endorf, C., Schultz, E., Mellander, J. *Detekce a prevence počítačového útoku*, Grada 2005, ISBN 80-247-1035-8
Dále podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Nevlud**

Datum zadání: 18.11.2011
Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 1. mája 2012 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 1. mája 2012 .....

Rád by som sa týmto poďakoval vedúcemu diplomovej práce Ing. Pavlovi Nevludovi za poskytnutie odborných informácií a materiálov, ktoré mi pomohli pri realizácii tejto diplomovej práce, taktiež mojím rodičom a blízkym, ktorý ma morálne podporovali a verili mi a v neposlednom rade chcem poďakovať priateľke.

Abstrakt

Firewall a systém prevencie proti prienikom môžu byť realizované ako samotný hardware alebo nasadené v podobe softvéru. Primárnym účelom týchto prvkov je pri firewalle určenie aká sieťová prevádzka je schopná prejsť. Zatiaľ čo systém prevencie proti prienikom odhaľuje nežiaducu aktivitu, ako napríklad neplatné pokusy pri pripojení sa na vzdialené služby alebo získavaní informácií o vnútornej siete, protokoloch, aplikáciách. Na udalosť reagujú na základe akcie, definovanej vnútornými pravidlami.

V rámci diplomovej práce sú popísané základné princípy firewallu a prevencie proti prienikom. Zmienené sú rôzne typy prekladu adries a portov na firewalle a rôzne analýzy pri zachytení dát, postupy pri tvorbe vlastných pravidiel alebo formáty výstrah. Popísaná je inštalácia jednotlivých služieb bežiacich v navrhnutej sieti, kde sa dbalo na bezpečnosť komunikácie, tvorba pravidiel na firewalle a systéme proti prienikom. K overeniu funkčnosti a poskytovanej ochrany bolo prevedených niekoľko typov útokov cez penetračné nástroje.

Klíčová slova: IPS, IDS, Snort, iptables, Firewall, pravidlá

Abstract

Firewall and intrusion prevention system can itself be implemented as hardware or software deployed in the form. The primary purpose of these is the firewall determine how network traffic is able to pass. While intrusion prevention system reveals undesirable activity, such as unsuccessful attempts to connect to remote service or obtaining information about internal networks, protocols, applications. At an event in response to the actions defined by internal rules.

The thesis describes the basic principles of firewall and intrusion prevention system. Mentioned are the different types of address translation and port on the firewall and various analyzes of the data capture, procedures for making their own rules and formats of alerts. Describes the installation of services running in the proposed network, which is striving for safety communication, create rules on the firewall and intrusion prevention system. To verify the functionality and protection provided was transferred to several types of attacks by penetrating instruments.

Seznam použitých zkratek a symbolů

IDS	–	Intrusion detection system
IPS	–	Intrusion prevention system
HIDS	–	Host-based intrusion detection system
NIDS	–	Network intrusion detection system
NIPS	–	Network Intrusion prevention system
GNU GPL	–	GNU General Public License
ISO	–	International Organization for Standardization
OSI	–	The Open Systems Interconnection
TCP	–	Transmission Control Protocol
IP	–	Internet Protocol
FTP	–	File Transfer Protocol
SID	–	Snort Identity document
UDP	–	User Datagram Protocol
ICMP	–	Internet Control Message Protocol
ARP	–	Address Resolution Protocol
IGRP	–	Interior Gateway Routing Protocol
GRE	–	Generic Routing Encapsulation
OSPF	–	Open Shortest Path First
RIP	–	Routing Information Protocol
IPX	–	Internetwork Packet Exchange
CIDR	–	Classless Inter-Domain Routing
DNS	–	Domain Name System
SMB	–	Server Message Block
DMZ	–	Demilitarized zone
ACL	–	Access control list
NAT	–	Network address translation
IOS	–	Internetwork Operating System
HTTP	–	Hypertext Transfer Protocol
SSH	–	Secure Shell
IPv6	–	Internet Protocol version 6

Typografická konvencia

Polotučné	–	obecný názov, pojem, parameter skriptu.
Index	–	anglický výraz.
Bezpatkové	–	dôležitá časť textu, príklad.
`Strojopisné`	–	názov súboru, modulu, skriptu, atribut v tabuľke, funkcie a vybraných programov.
Podčiarknuté	–	názov databázovej tabuľky.

Obsah

Zoznam tabuliek

Zoznam obrázkov

1 Úvod do problematiky IDS/IPS systémov

Existuje nespočetné množstvo definíc, ktoré popisujú narušenie, ale ja budem pracovať s touto definíciou [1.1].

Defínicia 1.1 *Narušenie je aktívna postupnosť odpovedajúcich udalosti, ktoré sa zámerné snažia uškodiť takou mierou, že popisovaný systém je nepoužiteľný, dochádza k sprístupnení neautorizovaných informácií alebo je s nimi manipulované.*

Predovšetkým si treba dávať pozor, pretože väčšina spôsobených problémov s detekciou narušenia je spôsobené nesprávnou implementáciou a nesprávnym chápaním toho, čo táto technológia môže a čo nemôže robiť.

1.1 Základy IDS a IPS

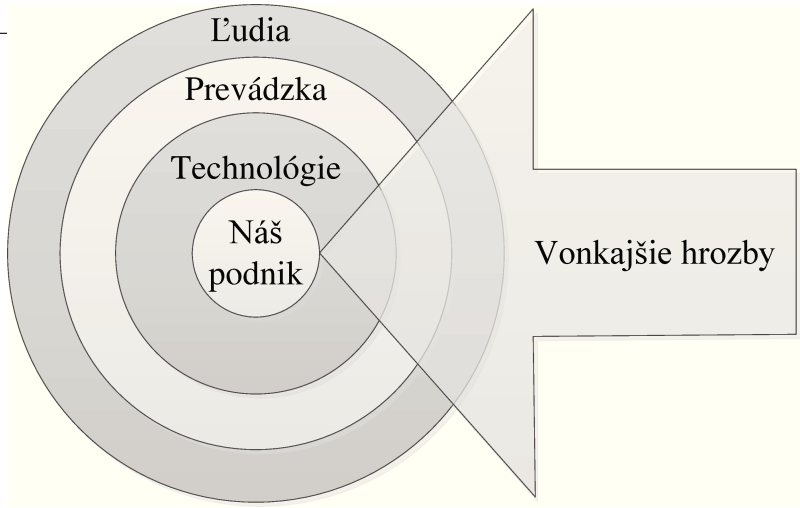
1.1.1 Čo je to detekcia narušenia

IDS môže byť definovaný ako súbor nástrojov, metód a zdrojov, ktoré nám pomáhajú identifikovať, sprístupniť a hlásiť neautorizované a nehlásené sieťové aktivity. Pričom IDS nedetekuje narušenie, ale detekuje také aktivity, ktoré môžu ale nemusia byť narušeniami. Detekcia narušenia je časťou celkového ochranného systému.

Firewall je možné voľne prirovnať k zamknutým dverám, detekciu narušenia k alarmovému systému a prevenciu proti narušeniu k strážnym psom. Zamknuté dvere zastavia nepovolaného jednotlivca, ale samy o sebe nás nijako neinformujú o narušiteľovi, iba zabránia neautorizovanému prístupu. Alarmový systém nás dokáže varovať pred potenciálnym narušením, avšak sám o sebe nezabráni samotnému narušeniu. Strážny pes, je v istých prípadoch schopný zabrániť narušeniu tým, že skríži plány narušiteľovi skôr ako sa dostane do chráneného objektu.

Zamknuté dvere, alarmový systém a strážny pes hrajú oddelenú, avšak doplňujúcu úlohu pri ochrane objektu. Taktiež to platí v prípade firewallu, IPS a IDS. Všetko sú to rôzne technológie, ktoré môžu spolupracovať a tým nám poskytnú ako vyrozumenie o narušeniu, tak prevenciu proti nemu. Efektívna stratégia ochrany objektu môže spočívať v tom, že alarmy a zámky umiestnime na všetky okna a dvere, a do vnútra objektu umiestnime detektory pohybu, možno umiestnenie niekoľkých strážnych psov vo vnútri obvodu.

IDS a IPS sú dve z mnoho metód, ktoré by mali byť využité v silnom bezpečnostnom programe. V závislosti na analýze rizík je rozhodujúce, aby bolo naše riešenie rozdelené do vrstiev alebo do hĺbky. Sieť by mala mať niekoľko bezpečnostných vrstiev s vlastnou funkcionalitou, ktoré tvoria celkovú bezpečnostnú stratégiu organizácie. Obrázok [1] ilustruje štruktúru hĺbkového riešenia, ktoré ochraňuje sieť na mnoho úrovniach.



Obr. 1: HĺbkovÁ ochrana

1.1.2 Typy IDS systémov

IDS spadá do troch základných kategórii:

1. Uzlovo orientované systémy detekcie narušenia (angl. *host-based intrusion-detection system*, **HIDS**) vyžadujú program, ktorý je umiestnený na tomto systéme a môže skenovať aktivitu všetkých uzlových zdrojov. Skenuje aktivity systémového a událostného logu. Zapíše ľubovoľnú udalosť do bezpečnostnej databázy a preverí, či sa tieto údaje nezhodujú so záznamami vadných udalostí obsiahnutých v databáze.

2. Sieťovo orientované systémy detekcie narušenia (angl. *network-based intrusiondetection system*, **NIDS**) sa zaradzuje do siete sériovo a analyzuje pakety, z čoho potom - usudzuje napadnutie. Všetky pakety príjme v zvláštnom segmente siete, vrátane prepínaných sieti (kde to nie je implicitné chovanie), pomocou jednej z metód ako napr. vetvenie alebo zrkadlenie (angl. *mirroring*) potrov. Starostlivo rekonštruuje prevádzkový sťový prúd a analyzuje v ňom prítomnosť závažné chovanie. Väčšina NIDS systémov je vybavená schopnosťou zaznamenávať súčinnosť, hlásiť alebo generovať výstrahu pri sporných prípadoch. Tieto schopnosti dnes ponúka väčšina vysoko výkonných smerovačov.

3. Hybridný IDS kombinuje HIDS, ktoré monitorujú udalosti odohrávajúce sa na uzlovom systéme, s NIDS, ktoré monitorujú sťovú prevádzku. Tabuľka [1] na strane 64 ukazuje základné rozdiely medzi NIDS a HIDS.

1.1.3 Čo je to systém prevencie proti narušeniu

Systém prevencie proti narušeniu (angl. *Intrusion-Prevention System*, **IPS**) je umiestnený sieti, ktorú monitoruje. Pokiaľ sa udeje nejaká udalosť, prijme sa opatrenie podľa predpísaných pravidiel.

Je to odlišné ako v prípade IDS, ktorý sa neumiestňuje sériovo do siete a je pasívny. Niektorý však považujú IPS za systém IDS budúcej generácie, pretože k detekcií dochádza v ďalšom kroku, avšak IPS je v skutočnosti odlišný bezpečnostný produkt líšiaci sa vo funkcionalite a sile, ktorý sa vyvinul z IDS.

Věta 1.1 ***Zber dát*** *- existujú dva základné spôsoby na prepínaných sieťach: zrkadlenie portov a vetvenie siete. Zrkadlenie portov (angl. port mirroring) sa taktiež nazýva premostenie (angl. spanning) a spočíva v tom, že sa prichádzajúce a odchádzajúce pakety skopírujú a predaju z jedného portu prepínača na iný port, kde môžu byť analyzované. Druhym spôsobom je sieťové vetvenie (angl. network taps), ktoré sa kladie sériovo do siete, skopírujú sa prichádzajúce a odchádzajúce pakety a znovu sa prenesú späť na tuto sieť.*

Systémy IPS so svojím nastavením podobajú IDS systémom, IPS môžu byť uzlovo orientované IPS (HIPS), ktoré najlepšie pracujú v ochranných aplikáciách alebo sieťovo orientované IPS (NIPS).

Užívateľské činnosti by mali odpovedať činnostiam v preddefinovaných znalostných databázach. Pokiaľ nejaká činnosť nie je v akceptovanom zozname, IPS jej zabráni uskutočniť. Logika sa v IPS aplikuje pred tým, než je prevedená v pamäti, na rozdiel od IDS.

Typické IPS sa skladá zo štyroch častí:

- Normalizátor prevádzky - prerušuje sieťovú prevádzku, prevádza analýzu a znovu zloženie paketov[1]
- Monitor služieb - vytvára referenčnú tabuľku, ktorá túto informáciu klasifikuje a napomáha tvarovaniu prevádzky riadiť tok informácií.
- Detekčná jednotka - porovnáva signálové vzorky s referenčnou tabuľkou a stanovuje príslušnú odpoveď.
- Prevádzková tvarovacia časť (tvarovač)

[1] a taktiež základné blokovacie funkcie.

1.1.4 IDS verzus IPS

Tieto dve odlišné technológie ako IDS, tak aj IPS majú svoje miesto v bezpečnostnom programe, pretože prevádzajú odlišné funkcie. Tabuľka [2] na strane 66 vyjadruje niektoré rozdiely medzi nimi.

1.2 Architektúra systémov IDS a IPS

Architektúra je jedná z najkritickejších aspektov v detekcií a prevencií proti narušeniu. Najefektívnejšia je taká, v ktorej každý stroj, zariadenie, komponenta a proces prevádzajú svoju rolu efektívnym a koordinovaným spôsobom, vyplývajúcim z účelného spracovania informácií, výstupov a taktiež príslušnej preventívnej odozvy, ktorá naplňuje obchodné a prevádzkové požiadavky organizácie.

Architektúra systémov detekcie a prevencie proti narušeniu sa zakladá na troch základných typoch vrstvových architektúr.

1.2.1 Jednovrstvová architektúra

Jednovrstvová architektúra je najjednoduchšia z architektúr, v ktorej komponenty zbierajú a spracovávajú dáta samostatne, bez toho aby ich predávala ako svoj výstup k spracovaniu iným skupinám komponent. Príkladom jednovrstvovej architektúry je na uzlu založený nástroj pre detekciu narušenia, ktorý tento výstup prijíma zo systémového logu a porovnáva ich so známymi vzorkami útokov.

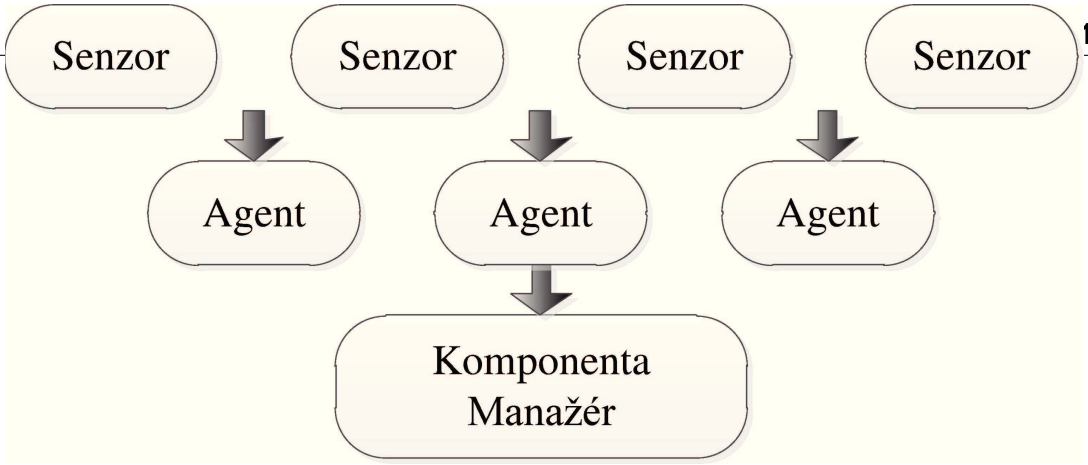
Tato architektúra ponúka výhody v podobe ako jednoduchosť, nízka cena (aspoň v prípade freeware bežiaceho na jednotlivých uzloch) a nezávislosť na iných kompotentách. V súčasnosti sa táto architektúra skladá z komponent, ktoré o sebe vzájomne nevedia, čo výrazne redukuje potenciálnu účinnosť a sofistikovanú funkčnosť.

1.2.2 Viacvrstvová architektúra

Zahrňuje celú radu komponent, ktoré si medzi sebou predávajú informácie. V súčasných systémoch IDS sú obsiahnuté tri primárne komponenty, ako je naznačene na obrázku [2]:

Senzory prevádzajú zber dát. Sieťové senzory sú často programy, ktoré zachytávajú dáta zo sieťového rozhrania. Môžu taktiež zbierať dáta zo systémových logov a ostatných zdrojov, napr. firewallov a rôznych komponent zapuzdrujúcich TCP.

Agenti (niekedy taktiež *analyzátory*) získavajú informácie zo senzorov a monitorujú aktivitu narušiteľov vo svojich vlastných uzloch. Každý senzor a agent je nakonfigurovaný tak, aby mohol bežať vo zvláštnom prostredí, v ktorom je umiestnený. Agenti sa špecializujú na vykonávanie jednej funkcie. Jeden agent môže monitorovať napr. TCP spojenie, zatiaľ čo druhý iba FTP spojenie a pokusy o neho. Funkcie ide rozširovať o nástroje tretích strán napr. nástroje pre monitorovanie sieti, neurónové siete a nástroje pre sledovanie spojenia.



Obr. 2: Viacvrstvová architektúra

Manažér získava informácie o napadnutiu alebo o práve prebiehajúcom útoku od agenta. Manažér môže vykonať celú radu funkcií vrátane nasledujúcich:

- zber a zobrazenie výstrah na konzole
- spustenie stránkovača alebo vytvorenie čísla celulárneho čísla
- uloženie informácie o udalosti do databázy
- vrátenie dodatočných informácií tykajúcich sa udalosti
- zapísanie informácie do uzlu, ktorý zastavil svoju činnosť kvôli spusteniu istých príkazov v pamäti.
- vyslanie príkazov na firewall alebo smerovač, aby zmenil riadiaci prístupový zoznam
- poskytnutie riadiacej konzoly

Výhody zahrňujú väčšiu účinnosť a hĺbku analýzy. Každá komponenta vykonáva funkciu, na ktorú bola navrhnutá a býva často nezávislá na ostatných. Poskytuje vysoký stupeň účinnosti, komplexnejší obraz bezpečnostných podmienok celej organizácie a ich vnútorných uzlov.

Hlavnými nevýhodami sú narastajúca cena a zložitosť. Množstvo komponent, rozhraní a komunikačných metód predstavuje zložitosť v konfigurácií tejto architektúry a náročnejšia údržba a riešenie problémov.

1.2.3 Architektúra peer-to-peer

Architektúra peer-to-peer zahrňuje výmenu informácií o detekcií alebo prevencií medzi rovnocennými komponentmi, z ktorých každá prevádza rovnaký druh činnosti. Často sa využíva pri spolupracujúcich firewalloch (a v menšej miere medzi spolupracujúcimi smerovačmi a prepínačmi). Pretože jeden firewall obsahuje informácie o práve sa odohrávajúcich udalostiach, predá je ďalšiemu firewallu, čo môže spôsobiť zmenu v riadiacom prístupovom zozname alebo pridanie reštrikcií na sprostredkované spojenia. Druhý firewall rovnako môže zaslať informácie, ktoré ovplyvnia chovanie prvého. Žiadny z oboch firewalloch nejedná ako centrály server alebo nadriadené úložisko informácií.

Hlavnou výhodou je ich budúcnosť. Akýkoľvek rovnocenný partner sa môže účastniť toho, čo je efektívne a môže profitovať z informácií zhromaždených ostatnými.

Nedostatkom je chýbajúca sofistikovaná funkcionalita kvôli absencií špecializovaných komponent, nakoľko je lepšia ako ta dosiahnuteľná v jednovrstvovej architektúre.

Architektúra peer-to-peer je vhodná pre organizácie, ktorých investície smerovali predovšetkým do obstarania vzájomne spolupracujúcich firewallov, avšak neinvestovali toľko do systémov IDS a IPS. Najlepšími zdrojmi dát detekcie narušenia sú firewally. Pokiaľ tieto získane dáta zdieľame a distribuujeme medzi rovnocennými partnermi a potom na tomto základe činíme zmeny v riadiacich prístupových zoznamoch alebo pravidlách pre proxy, môže to do istej miery kompenzovať chýbajúce IDS a IPS.

2 SNORT

Tento nástroj je multiplatformný a voľne šíriteľný pri dodržaní licencie GNU GPL v.2. Snort je možné použiť v kategórií IPS i IDS. Je známy ako NIDS, pričom ak je nakonfigurovaný k analýze prevádzky smerovaného do a z jednotlivého uzlu, môže byť použitý ako HIDS. Jeho detekčná jednotka závisí na rudimentárnom jazyku[2]. Poskytuje rýchly a ľahký prístup k dátam primárneho paketu, čo napomáha k presnejšej inšpekcií sieťovej prevádzky alebo záznamu toho, čo Snort našiel [2].

Snort povodne nebol vyvíjaný ako koncové IDS, ale ako jednoduchý a pružný doplnok, ktorý by s IDS bežal súčasne. Na Snort sú kladené požiadavky:

- z dôvodu výkonu minimalizovať rozhranie medzi systémom a sieťou
- prevádzkovať ho v rôznych častiach siete,, zbierať dáta, ktoré môžu byť zhromaždené v jednom bode
- optimalizovať pomernú početnosť a naopak minimalizovať početnosť falošných poplachov
- ponúknuť jednoduchšie obsluhovateľné funkcie pre generovanie zostáv
- detekovať incidenty a generovať výstrahy v reálnom čase
- poskytovať schopnosti paketového záchytu[3]

2.1 Režimy Snortu

Snort beží v troch rôzných režimoch, ktoré vám predstavím v nasledujúcej časti.

2.1.1 Režim sliediča

Režim sliediča (angl. *sniffer*) umožňuje zachytávať dáta v hlavičke každého paketu na obrazovku. Činnosť Snortu a dáta zobrazíme zadaním príkazu:

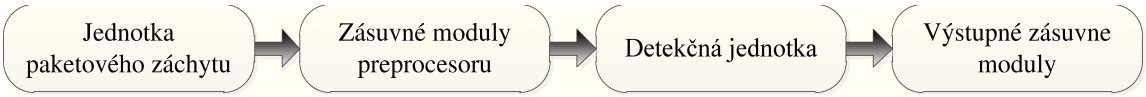
```
./snort -dv
```

Je možné pomocou voľby prepínača zobrazovať na obrazovke rôzne vrstvy referenčného modelu ISO/OSI[4], ako napr. prepínač `-v` zobrazí na obrazovke 3. a 4. vrstvu ISO-/OSI modelu, prepínač `-d` aplikačné dáta a prepínač `-e` zobrazí spojovú vrstvu.

[2] Slúži ako základ pre rozpoznávanie vstupných vzorov charakteristických pre útoky.

[3] Ako je zmienené v kapitole o architektúre systémov IDS a IPS na strane 9, je schopnosť rýchleho záchytu kritickou funkciou IDS.

[4] Referenčný model ISO/OSI sa používa ako názorný príklad riešenia komunikácie v počítačových sieťach pomocou vrstevnatého modelu, kde sú jednotlivé vrstvy nezávislé a ľahko nahraditeľné [3]



Obr. 3: Štyri komponenty IDS Snort

2.1.2 Režim záznamníku

Režim záznamníku (angl. *logger*) na rozdiel od sliediča zapisuje zachytené dáta na pevný disk uzlu do adresára `log` a to nasledovným príkazom:

```
./snort -dev -l ./log -h 10.0.0.0/24
```

Prepínač `-d` a následný argument sú nepovinné, ale ktoré zaistia, že Snort zachytáva dáta s cieľovou adresou siete `10.0.0.0/24`. Pre každú adresu sa v adresári `log` vytvorý oddelený podadresár. Na vysokorýchlostnej sieti postačuje jediný prepínač `-b`, ktorý spôsobí binárny výstup do jednoho adresára, pričom ostatné prepínače `-d -e -v` nie sú potrebné. Prezeranie jedného typu dát napr. TCP docielime zadaním:

```
./snort -dvr packet.log tcp
```

2.1.3 Režim sieťového detektoru narušenia

Predchádzajúce dva režimy nástroja Snort sú vhodné pre veľkokapacitný záchyt dát, preto Snort v sieťovom detekčnom režimu narušenia nezaznamenáva všetky dáta, ale skôr umožňuje vybrať dáta podľa definovaných pravidiel. Súbor `snort.conf` implicitne tieto pravidlá obsahuje. Základným spôsobom prevádzkovania Snortu v režimu detekcie narušenia je zadanie:

```
./snort /dev /l ./log -h 10.0.0.0/24 -c snort.conf
```

2.2 Komponenty

Snort sa vyznačuje štyrmi hlavnými komponentmi IDS: jednotka paketového záchytu, zásuvný modul preprocesoru, detekčná jednotka a zásuvný modul pre výstup, ako je znázornené na obrázku [3].

2.2.1 Jednotka paketového záchytu

Jednotka paketového záchytu (angl. *packet capture engine*) zbiera prevádzkové dáta pomocou knižníc `libpcap` alebo `WinPcap` (na obe sa budem odkazovať zjednodušene `pcap`).

Táto knižnica `pcap` umožňujú aplikáciám prijímať rámce, ktoré fyzický zachycuje sieťová karta a predáva ich ovladaču rozhrania jádra OS. Keď jadro spracuje tieto dáta, knižnica `pcap` od neho prevezme dáta a predá ich aplikáciám Snortu, ktoré sú naviazané

na zásuvné moduly preprocesoru. Ak neexistujú žiadne interpretery - zásuvné moduly, `pcap` zašle všetky dáta do príslušných aplikácií, ktoré ich musia filtrovať, aby nimi neboli zaplavené.

2.2.2 Zásuvné moduly preprocesoru

Zásuvné moduly preprocesoru testujú a prehľadávajú dáta prijaté z `pcap`, určujú, čo previesť s každým paketom, či ho analyzovať, zmeniť, odmietnuť alebo vygenerovať výstrahu kvôli jeho obsahu.

Preprocesory modifikujú URI a URL, aby odpovedali štandardnému formátu, prevádzajú stavovú analýzu prevádzky TCP/IP, detekujú skenery portov, dekódujú pakety RPC a telnet atď. Pokiaľ zásuvné moduly preprocesoru daný vstup neodmietnu, predajú ho nasledujúcej komponente, detekčnej jednotke.

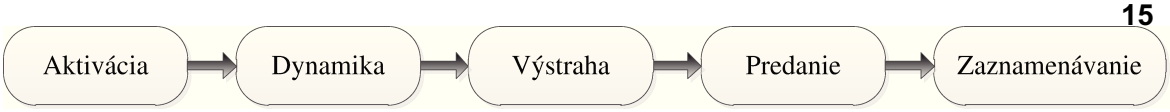
2.2.3 Detekčná jednotka

dáta sú najprv dekódované spôsobom, ktorý stanový ich štruktúru pre protokoly druhej a tretej vrstvy. To umožňuje detekčnej jednotke systematický porovnávať dáta vo vnútri každého paketu, ktorý podľa zvoleného pravidla prijala. Táto jednotka prevedie test, či obsahuje zvláštny reťazec alebo hodnotu združenú s nejakým pravidlom a potom spustí ďalší test s použitím nasledujúceho pravidla a tak ďalej, až nie sú otestované všetky pravidlá. Potom sa detekčná jednotka presunie k spracovávaniu ďalšieho paketu.

Rôznorodosť zásuvných modulov umožňuje ich použitie v mimoriadnych detekčných testoch. Každá voľba kľúčového slova v každom pravidle je združená s nejakým zásuvným modulom.

2.2.4 Zásuvné moduly pre výstup

Hlavným účelom zásuvných modulov pre výstup je vytvoriť informáciu, ktorá sa zobrazí v analýze detekcie narušenia. Snort vytvára výstrahy v závislosti na výstražných pravidlách preprocesorov, dekódovacích jednotiek a detekčného stroja [2].



Obr. 4: Reťazec implicitného radenia aplikovaných pravidiel v Snortu

2.3 Pravidlá Snortu

Využitie pravidiel Snortu je srdcom jeho funkcionality ako IDS. obecný tvar pravidla Snortu je nasledovný:

```
akcia protokol IP_zdroj port_zdroj smer IP_ciel port_ciel (moznosti)
```

2.3.1 Povaha pravidiel

Medzi mnohými typmi pravidiel sú, ktoré sa týkajú FTP a webových útokov, distribuovaných útokov DoS, pokusov o zneužitie zraniteľnosti v protokolu SMB v systémoch Windows a útokov na vzdialené volanie procedúr RCP. Každé pravidlo korešponduje s nejakým číslom, identifikátorom senzoru (angl. *Snort Identification*, **SID**), ktoré komukoľvek umožňuje ľahko odlíšiť jedno pravidlo od druhého.

Postupom času sa podstatne zvýšil počet dostupných pravidiel, pretože bola vyvinutá rada metód zneužitia. V dôsledku toho sa začali pravidla organizovať na základe jeho typu v určitom rozsahu hodnôt SID. Je možnosť si vybrať medzi použitím všetkých týchto pravidiel alebo ich ľubovoľnú podmnožinu. Ak užívame celú množinu, získame komplexnejšiu schopnosť detekcie, avšak za cenu zvýšenia výkonu. Čím viac pravidiel Snort spracúva, tým pomalšie pracuje.

2.3.2 Písanie pravidiel

V Snorte je možné si vytvoriť užívateľské pravidlá, ktoré nie sú súčasťou sady pravidiel. Každé pravidlo sa skladá z dvoch častí:

- z hlavičky - obsahuje činnosť spojenú s týmto pravidlom (Výstraha, Záznam, Predanie, Aktivacia alebo Dynamika alebo užívateľské pravidlo), typ protokolu, zdrojovú a cieľovú IP adresu, masku a zdrojový a cieľový port.
- z voľby pravidla - obsahuje výstražnú správu a informáciu týkajúcu sa časti každého paketu, ktorý bude analyzovaný, aby mohlo byť rozhodnuté, či bude uskutočnená špecifikovaná činnosť.

2.3.3 Hlavička

Detekčná jednotka Snort vyšetruje reťazce piatich pravidiel: Aktivácia (angl. *Activation*), Dynamika (angl. *Dynamic*), Výstraha (angl. *Alert*), Predanie (angl. *Pass*) a Zaznamenávanie (angl. *Log*), ako je vidieť na obrázku [4].

Pokiaľ je Snord prevádzkovaný v režimu inline, je ponuka akcií rozšírená o tri pravidlá:

- zahodenie (angl. *drop*) - blokovanie a logovanie paketu
- odmietnutie (angl. *reject*) - ako zahodenie, ale druhej strane zašle upozornenie o zahodení
- sdrop - blokovanie paketu bez záznamu

Radenie týchto pravidiel do postupnosti ide meniť. Snort zahajuje s plnou záznamovacou schopnosťou (ak zapneme prepínač `-l`) a v tom prípade implicitne načíta zo súboru `snort.conf` s aplikovanými pravidlami. Táto notácia znie:

```
./snort -ll ./log -c snort.conf
```

Prepínač `-l` znamená použitie plného zaznamenávania a `./log` znamená záznam do súboru `log` v aktuálnom adresári. Príznak `-c` predstavuje pokyn Snortu, aby načítal konfiguračný súbor.

Ak Snort aplikuje reťazec pravidiel, prejde do uzlu pravidlového stromu, kde sa aplikujú ďalšie špecifické protokolové pravidlá. Uzly stromu pravidiel (angl. *Rule Tree Nodes*) analyzujú TCP, UDP, ICMP a IP prevádzku. V každom uzle ležia voľby, jedná pre Obsah (angl. *Content*), ktorá sa premieta do algoritmu aplikácie porovnávania vzorov na paketové dáta, a druhá pre Prietok (angl. *Flow*), ktorej dôsledkom je aplikovaný zásuvný modul pre detekciu.

Protokol V súčasnej dobe existujú štyri protokoly, ktoré Snort analyzuje za podozrivé správanie:

- TCP
- UDP
- ICMP
- IP

Do budúcnosti sa počíta ešte s ARP, IGRP, GRE, OSPF, RIP, IPX atď.

IP adresy Ďalšou časťou z hlavičky pravidla sú IP adresy a porty. Slovo *any* definuje akúkoľvek adresu. Adresy sú tvorené priamo číselnou IP adresou v CIDR notácii [4].

V nasledujúcom príklade je použitá ľubovoľná IP adresa a cieľová `192.168.1.0/24`.

```
alert tcp any -> 192.168.1.0/24 111 \
(content:"|00␣01␣86␣a5|"; msg:"external␣mountd␣access";)
```

Pričom je možné použiť operátor negácie *"!"* pred IP adresu, ktorý by rozlišoval vnútornú a vonkajšiu sieť.

Čísla portov Čísla portov môžu byť špecifikované viacerými spôsobmi, vrátane všetkých portov *any*, číselných, rozsahu portov, a negácie v obecne platnom rozsahu portov 0 - 65535.

Záznam UDP prevádzky prichádzajúcej z neurčitej IP adresy a portu na cieľový port v rozsahu od 1 do 1024.

```
log udp any any -> 192.168.1.0/24 1:1024
```

Rovnaký záznam prevádzky prichádzajúci na port menší alebo rovný 6000.

```
log tcp any any -> 192.168.1.0/24 :6000
```

Záznam z portu menšieho alebo rovnakého ako 1024 na port väčší alebo rovný 500.

```
log tcp any :1024 -> 192.168.1.0/24 500:
```

Smer prevádzky Operátor smeru > označuje orientáciu alebo smer prevádzky. Môže sa použiť aj operátor <>, ktorý využívame, keď potrebujeme použiť pravidlo v oboch smeroch prevádzky, napr. pre telnet alebo POP3.

```
log tcp !192.168.1.0/24 any <> 192.168.1.0/24 23
```

2.3.4 Voľba pravidiel

Všetky voľby pravidiel sú od seba oddelené pomocou bodkočiarky *;* a kľúčové slova od svojich argumentov dvojbodkou *:*.

Existujú štyri hlavné kategórie voľby pravidiel [2]:

- všeobecné (angl. *general*) parametre bývajú užité v každom z pravidiel, niektoré sú priamo vyžadované, ako napr. *sid*
  - *msg:«message text>";* - hlásenie pre výstrahu a logovanie paketu
  - *reference:<id system>, <id>;* - popisuje, kde ide nahľadať informácie o danej signatúre
  - *gid:<generator id>;* - identifikácia kompotenty, ktorá výstrahu generovala
  - *sid:<snort rules id>;* - jedinečný identifikátor pravidla
  - *rev:<revision integer>;* - revízia pravidla
  - *classtype:<class name>;* - zarad'ovanie do tried udalostí podľa *classification.config*
  - *priority:<priority integer>;* - nastavenie priority pravidlu
  - *metadata:key1 value1;* - nastavenie priority pravidlu

- (angl. *payload*) - pomocou týchto parametrov ide kontrolovať, či paket prenáša hľadanú štruktúru.
  - *content:[!]«content string>";* - hľadanie zadaného reťazca
  - *nocase;* - nerozlišuje malé a veľké znaky
  - *rawbytes;* - deaktivuje dekódovanie a vyhodnocuje surový obsah paketu
  - *depth:[<number>|<var_name>];* - definuje hĺbku hľadania
  - *offset:[<number>|<var_name>];* - posunutie začiatku hľadania
  - *dsize:min<>max;* - veľkosť dátového paketu
  - *uricontent:[!]«content string>";* - prehľadáva URI požiadavky
  - *urilen:min<>max[,<uribuf>];* - definuje dĺžku URI požiadavku
- (angl. *non-payload*) - nasledujúce parametre umožňujú detekovať pole príznakov, nesené hlavičkou paketu.
  - *ttl:[<, >, =, <=, >=]<number>;* - doba životnosti paketu
  - *flags:[!|*|+]<FSRPAUCE0>[,<FSRPAUCE>];* - nastavené príznaky v poli Flags
  - *tos:[!]<number>;* - hodnoty nastavené v poli TOS
  - *ack:<number>;* - sekvenčné čísla potvrdzovacích paketov
  - *itype:min<>max;* - špecifický typ ICMP správy
  - *window:[!]<number>;* - nastavenie veľkosti okna prijatého TCP paketu
- (angl. *post-detection*) - vo vnútri pravidla je možné definovať dodatočné akcie, ako napr. zaznamenávanie konkrétnej udalosti do samotného súboru, nahradenie nájdeného reťazca alebo aktivácia dynamického pravidla.
  - *logto:"filename";* - zaznamenávanie udalosti do samotného súboru
  - *session:[printable|binary|all];* - umožňuje získať informácie o danom TCP spojení
  - *replace:«string>";* - v režime inline nahradzuje hľadanú hodnotí inou
  - *resp* - v prípade výstrahy ukončuje TCP reláciu
  - *react* - ukončuje spojenie a vystavuje výstrahu
  - *activates:1;* - aktivácia dynamického pravidla
  - *activated_by:1;* - určenie aktivačného zdroja v dynamických pravidlách
  - *activated_by:1; count:50;* - počet prepustených paketov pred aktiváciou dynamického pravidla

2.3.5 Filtre v Snorte

Filtre napomáhajú pri riadení objemu prevádzky, ktorá musí spracovávať. Filtre určujú čo sa bude zachytávať. Napríklad ak sa ma zachytiť DNS a UDP prevádzka na porte 53, nadefinujeme filter takto:

```
udp and dst port 53
```

pričom filter aktivujeme príkazom `tcpdump`.

2.4 Výstup

Funkcia Snortu dovoľuje zaslanie výstražných dát ako zachytené SNMP dáta alebo pripojenie do záznamového súboru či databázy.

2.4.1 Výstrahy

Formaty výstrah Každá detekovaná položka je zobrazená v záznamovom súbore, ktorého oddeľovačom je dvoj hviezdička.

```
d[**] [1:2000100:2] SCAN SYN -sS [**]
[Classification: Detection of a Network Scan] [Priority: 3]
05/12-23:18:46.901831 192.168.0.2:43221 -> 192.168.0.1:1898
TCP TTL:59 TOS:0x0 ID:2028 IpLen:20 DgmLen:44
******S* Seq: 0x5DC53C2 Ack: 0x0 Win: 0x1000 TcpLen: 24
TCP Options (1) => MSS: 1460
```

Snort označil v tomto prípade oba TCP pakety ako časť prehliadky s paketami SYN-FIN zaslanými na cieľový uzol. Tieto pakety boli umelo vyrobené, o čom svedčí sekvenčné číslo ID a to, že i potvrdzovacie číslo sa nezmenilo ako aj zdrojový port, čo sa v reálnych prípadoch nestáva.

Poslanie paketu SYN/ACK bez predošlého SYN paketu je metóda narušenia, ktorá sa snaží paket pretlačiť skrz firewall, ktorý nepraktikuje stavovú analýzu prevádzky.

Režimy výstrah

-full - implicitný režim, v ktorom Snort tlačí obsah výstražných správ a celú paketovú hlavičku každého paketu

-fast - pre zjednodušený formát, ktorý obsahuje čas, samotnú správu, zdrojovú a cieľovú IP adresu a port

-socket - zasiela výstrahy do unixových socketov a umožňuje ich predať do programu

-syslog - zasiela výstrahy do protokolových záznamoch Unixu, Linuxu, Windowsu a taktiež umožňuje rôzne výstrahy zaslať na centrálnu konzolu

-smb - protokol SMB (angl. *Server Message Block*)[5] je použitý pri rozosielaní výstrah

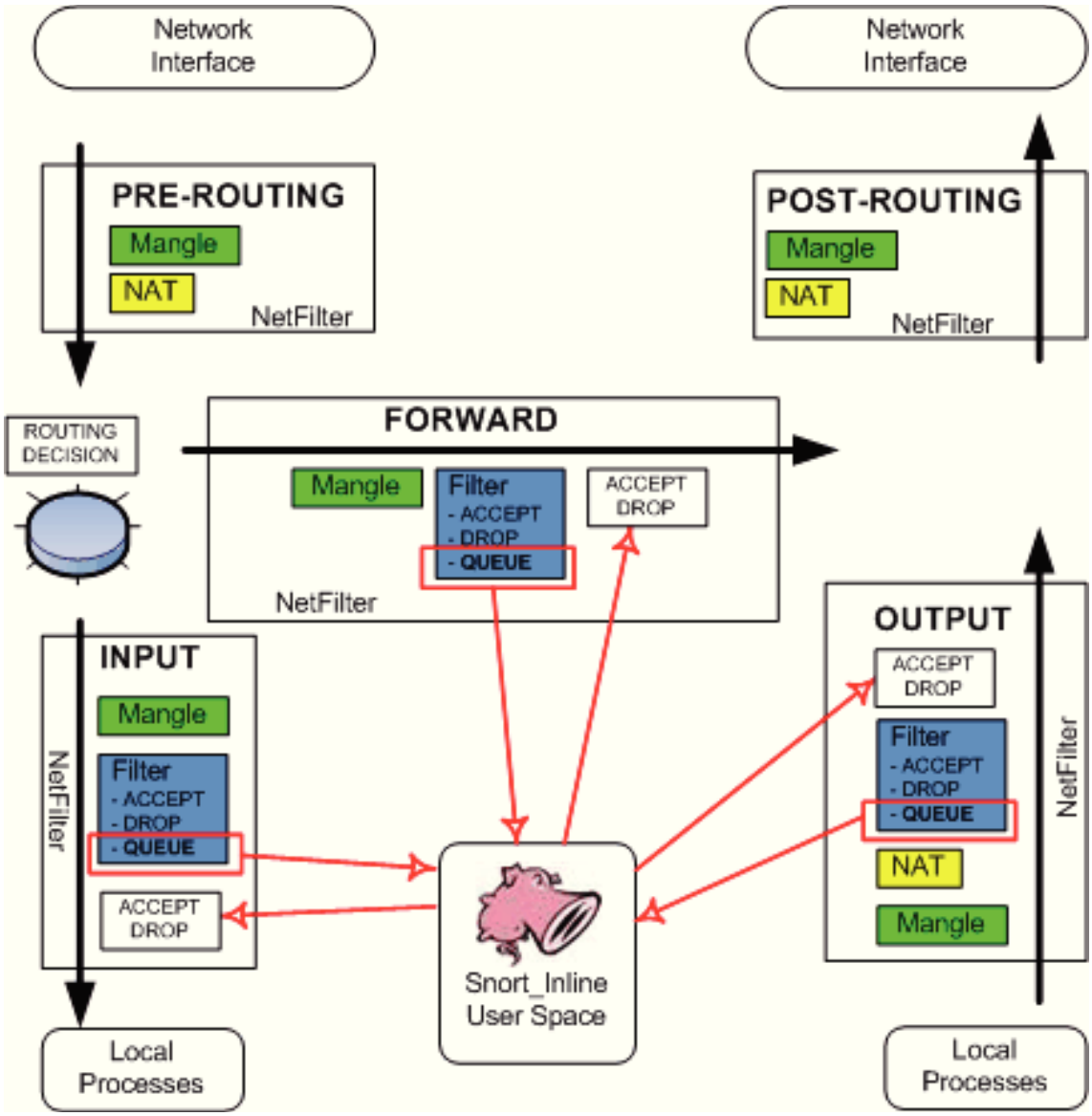
-none - vypínanie všetkých výstrah

2.5 Režim Inline

Základom prevádzky systému Snort v režime Inline je firewall implementovaný v OS Linux, ktorý obsahuje reťazce INPUT, OUTPUT a FORWARD. Obsahujú pravidlá vrátane akcie, pričom pri tomto režime Inline sa používa QUEUE - zaradenie do fronty.

Snort pracujúci v režime Inline vyzdvihuje pakety z front a porovnáva ich s vlastnými pravidlami, na ktorých prevádza jednu z definovaných akcií popísaných v kapitole 5.4.7 na strane 50. Jednoduchým príkladom môžu byť alert a drop, kde pri akcií alert dochádza k navráteniu paketu späť do reťazca a vydaní varovania. V prípade drop sú pakety zahodené. Vnútorné usporiadanie Netfilteru a Snortu zachytáva obrázok [5]. [2]

[5]Protokol využívaný Windows (a taktiež službou SAMBA v Linuxu) k prenosu informácií z jedného uzla na druhý



Obr. 5: Preberanie paketov z tabuliek Netfilteru

3 Čo je to FireWall?

Firewall je obecne bezpečný a dôveryhodný systém zapojený medzi privátnu - Intranet a verejnú sieť - Internet. Firewallový systém ma nastavené pravidla udávajúce, aky sieťový tok môže prepúšťať a aký ma byť zablokovaný alebo odmietnutý. V niektorých veľkých organizáciách sa používajú firewally aj vo vnútri siete ako ochrana citlivých oddelených častí organizácie od ostatných zamestnancov, napr. DMZ (angl. *Demilitarized Zone*).

Firewally je možné konštruovať rôznymi metódami, pričom najpokročilejšia používa niekoľko samostatných systémov a rozdeľuje sieť na rôzne zabezpečené úrovne. Tento typ ochrany sa používa obvykle vo veľkých spoločnostiach.

Typickejšie je však použitie firewallu jediným počítačom, ktorý zaisťuje všetko. Jedná sa o menej bezpečné riešenie, pretože pokiaľ bude v samotnom firewallu chyba, ktorá umožní neautorizovaný prístup k nemu, môže byť narušená celá bezpečnosť siete.

3.1 Činnosť firewallu

Firewall postupne zisťuje, či parametre prijatých sieťových dát vyhovujú pravidlám definovaných správcom siete. S dátami, ktoré podmienkam daného pravidla vyhovujú, prevedie akciu udanou tímto pravidlom [8].

3.1.1 Kategorie firewallu

3.1.2 Paketovy filter

Najjednoduchší a najstarši spôsob, kde pravidla presne udávajú z akej adresy a portu na akú adresu a port ma byť doručený predchádzajúci paket, t.j. kontrola je prevádzaná na tretej a štvrtej vrstve sieťového modelu OSI. Hlavnou výhodou je vysoká rýchlosť spracovania, preto sa ešte dnes môžeme s ním stretnúť tam, kde není potrebná presnosť alebo dôkladnejšia analýza prechádzajúcích dát, ale skôr ide o vysokorýchlostný prenos veľkého množstva dát. Na druhej strane je nevýhodou nízka úroveň kontroly prechádzajúceho spojenia, navyše pre umožnenie takéhoto spojenia vyžaduje otvoriť i porty a smery spojenia, ktoré môžu byť využite inými protokolmi.

Hlavní predstavitelia paketových filtrov patria ACL (*Access Control Lists*) v starších verziách operačného systému IOS na smerovačoch CISCO Systems, JunOS spoločnosti Jupiter Networks.

3.1.3 Aplikačné brány

Aplikačné brány, ktorým sa hovorí aj Proxy firewally na rozdiel od paketových filtrov úplne oddeľujú siete. Všetka komunikácia cez aplikačnú bránu prebieha formou dvoch

spojení - klient (iniciátor spojenia) sa pripojí na aplikačnú bránu (proxy), ktorá prichádzajúce spojenie spracuje a na základe požiadavky klienta otvorí nové spojenie k serveru, pričom klient predstavuje aplikačnú bránu. Aplikačná brána predá prichádzajúce dáta, ktoré dostala od serveru, v pôvodnom spojení klientovi. Kontrola je prevádzaná na siedmej (aplikačnej) vrstve sieťového modelu OSI.

Sever nevidí zdrojovú adresu klienta, ktorý je pôvodcom požiadavku. Ako zdroj je uvedená vonkajšia adresa aplikačnej brány, vďaka čomu pôsobia ako nástroje pre preklad adries NAT (*Network Address Translation*), čo je pomerne vysoké zabezpečenie známych protokolov.

Nevýhodou je vysoká náročnosť na použitý hardware. Aplikačné brány spracovávajú niekoľkonásobne nižšie množstvo spojení a rýchlosti ako paketové filtre, pričom majú omnoho vyššiu latenciu. každý protokol vyžaduje napísanie špeciálizovaného proxy, alebo využitie generického proxy, ktoré nie sú o nič bezpečnejšie ako využitie paketového filtra. V minulosti dokázali aplikačné brány kontrolovať okolo 10 protokolov, pričom pôvodné brány vyžadovali, aby klient vedel s aplikačnou bránou komunikovať a pritom nedokázali dobre chrániť svoj vlastný operačný systém. Po nástupe stavových aplikačných filtrov sa vývoj väčšiny aplikačných brán zastavil a tie, ktoré ostali sa používajú vo veľmi špecializovaných nasadeniach.

Predstaviteľmi aplikačných brán boli napr. The Firewall Toolkit (fwtk) a z neho vychádzajúci Gauntlet spoločnosti TIS neskôr zakúpený spoločnosťou NAI.

3.1.4 Stavové paketové filtre

Stavové paketové filtre prevádzajú kontrolu rovnako ako jednoduché paketové filtre, navyše si však ukladajú informácie o povolených spojeniach, ktoré potom môžu využiť pri rozhodovaní, či paket patrí do už povoleného spojenia a môže byť prepustený, alebo musí prejsť rozhodovacím procesom. To urýchľuje spracovanie paketov už povolených spojení a v pravidlách pre firewall umožňuje zahrnúť do povoleného spojenia i súvisiacu komunikáciu. Napríklad pre FTP (File Transport Protokol) stačí nastaviť pravidlo, v ktorom povolíme klientovi pripojenie sa na server pomocou FTP. Pretože sa jedná o známy protokol, firewall sám nastaví riadiace spojenie cez port 21 a taktiež aj port 20 pre dátovú komunikáciu. Umožňuje vytvoriť aj virtuálneho stavu spojenia pre bezstavové protokoly ako napr. UDP a ICMP.

K najväčším prednostiam stavových paketových filtrov patrí ich vysoká rýchlosť, slušná úroveň zabezpečenia v zrovnaní s vyššie spomenutými aplikačnými bránami a jednoduchými paketovými filtrami, ľahšia konfigurácia a nižšia pravdepodobnosť chybného nastavenia pravidiel obsluhou.

Oproti aplikačným bránam poskytujú stavové paketové filtre nižšiu bezpečnosť.

Typickými predstaviteľmi sú napr. FireWall-1 spoločnosti Check Point do verzie 4.0, staršie verzie Cisco PIX, Cisco IOS Firewall, staršie verzie firewallov Netscreen spoločnosti Jupiter a z voľne dostupných produktoch ipfw v BSD, či iptables v linuxovom jadre, o ktorých sa podrobnejšie budem venovať v ďalších kapitolách.

3.1.5 Stavové paketové filtre s kontrolou protokolov a IDS

Súčastné stavové paketové filtre okrem informácie o stavu spojenia a schopnosti dynamicky otvárať porty pre rôzne riadiace a dátové spojenia implementujú niečo, čo sa v markentigovej terminológii nazýva Deep Inspection alebo Application Intelligence. Tieto firewally dokážu prechádzať dáta známych protokolov a aplikácií a tak kontrolovať korektnosť dát. Pokiaľ napríklad objavia v http spojení indikátory, že sa nejedná o požiadavku na www server, ale o tunelovanie iného protokolu(často využívajú klienti p2p sieti), zakážu priechod.

Tieto systémy zaručujú vysokú úroveň bezpečnosti kontroly prechádzajúcich protokolov pri zachovaní ľahkej konfigurácie. Rýchlosť kontroly je v porovnaní s aplikačnými bránami vysoka, ale oproti stavovým paketovým filtrom stále o tretinu až polovicu spomalená.

Nevýhodou u týchto systémov je, že integrujú veľké množstvo funkcionality čo zvyšuje pravdepodobnosť zneužitia chyby v kóde.

Hlavnými predstaviteľmi tejto kategórie sú Check Point FireWall (od verzie 4.1, teraz NGX), produkty rady Netscreen, ISG a SSG spoločnosti Jupiter a v experimentálnom modulu taktiež pre iptables v linuxovom jadre [7].

3.2 Funkcie Firewallu

V súčasnosti neplní moderný firewall len základné funkcie ochrany pred únikom dát, či napadnutím lokálnej siete, ale prináša komplexné riešenie v oblastiach napojenia Internetu a lokálnej siete. Tieto služby dokážu plniť funkcie antivírusovej ochrany, optimalizáciu pripojenia, problémy s IP, prístupových práv užívateľov, zabezpečenie komunikácie, zdieľanie prístupu k Internetu apod.

Antivírusová ochrana Ďalšia služba, ktorá s firewalom tvorí symbiózu, a na ktorú sa kladú požiadavky na ochranu pred prichádzajúcimi a odchádzajúcimi počítačovými vírmi hlavne cez elektronickú poštu.

Optimalizované pripojenie Týmto sa väčšinou rozumie využívanie cache serverov. Tie majú za úlohu uchovávať často otvárané stránky, obrázky a iné zdroje z Internetu do pamäti a pri ďalšom požiadavku na ich otvorenie sa už načítajú zo servera umiestneného priamo na bráne firewallu. To ma za dôsledok, že si užívatelia vystačia aj s pomalším

pripojením do Internetu, pretože pri opakovanom pripojení na rovnakú službu sa otvorí už iba z lokálneho serveru.

Problémy s IP V dnešnej dobe ako je to už pri drvivej väčšine zariadení vo verejnej sieti, teda v Internete, tak je problém s prirad'ovaním verejných IP adries. Každý sieťový uzol potrebuje pre svoju lokalizáciu jedinečnú IP adresu, čo je mnoho krát nerealizovateľné a finančne náročné. Preto sa pri dnešných firewalloch stretneme s možnosťou NAT (Network Address Translation) prekladu adries z privátnych, nezávislých na poskytovateľovi Internetu, ktoré sú za firewallom na jedinú verejnú pridelenú poskytovateľom Internetu, s ktorou sa identifikujeme v Internete.

Prístupové práva Prevádzkovateľ podnikovej siete môže pomocou firewallu obmedzovať niektoré sieťové služby, ktoré nesúvysia s náplňou práce napríklad - prehliadanie si erotických stranok, používanie instant messengerov, sťahovanie cez rôzne služby atď. Ďalej je možné ľahko monitorovať služby a ukladať si ich do protokolu.

Zabezpečená komunikácia Pomocou dvojice firewallov ide prevádzať komunikáciu skrz Internetu na úrovni zabezpečeného tunelu. Taký tunel je praktický neviditeľný a nezávislý na topológii, cez ktorú prechádza. Dnes sa používa prevádzkovanie cez Internet ako VPN.

Zdieľanie prístupu k Internetu Firewall je možné v malých firmách využívať v koexistencií so smerovačom na jednom zariadení, kde toto zariadenie je vstupnou bránou do Internetu a zároveň zabezpečuje konektivitu všetkým pripojeným PC v privátnej sieti [6].

4 NETFILTER a IPTABLES

Nástroj iptables je súčasťou zdrojového balíku netfilter, ktorý vznikol ako podnet o zjednodušenie mechanizmu filtrovania datagramov vo firewallovom jadre, konkrétne v linuxu od jadra 2.4. Netfilter od predchádzajúceho ipchans rieši zložitosť a neobratnosť tak, že v jadre implementuje obecnú platformu, ktorá zjednodušuje proces spracovávania datagramov a zároveň umožňuje modifikovať filtračné politiky bez nútnosti upravovať jadro. V implementácií ipchains sa trieda input vzťahuje na všetky datagramy prijaté počítačom, bez ohľadu na to, či sú určené pre neho, alebo či ich má iba smerovať. Pričom trieda input v implementácií netfilter sa vzťahuje na datagramy určené lokálnemu hostiteľovi, a trieda forward potom iba na datagramy určené inému hostiteľovi. Anatomický to platí aj pre triedu output v oboch implementáciách.

Základným nástrojom pre nastavenie paketového filtra mechanizmu netfilter je známy riadkový nástroj iptables. Jeho syntax je odvodená od ipchains, pričom je rozšíriteľná. To znamená, že funkciu programu je možné doplniť bez nutností jeho nového preloženia, na základe zdieľaných knižníc [5].

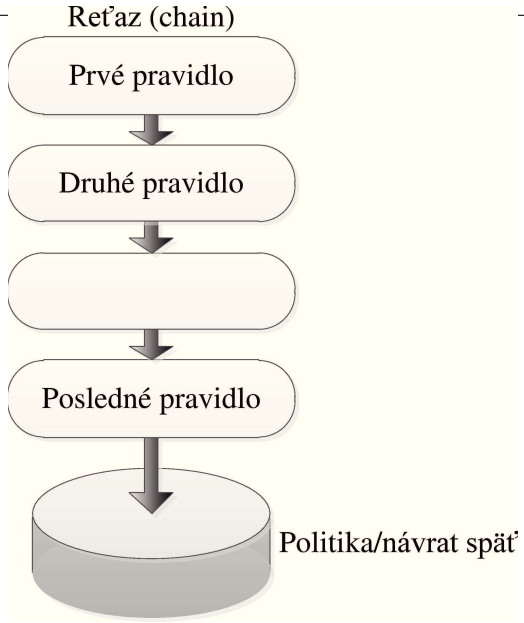
Netfilter sám o sebe nie je len firewall, ale taktiež plní hlavne úlohu paketového filtra, ktorý mimo iné môže vytvárať pravidlá potrebné pre funkcionalitu firewallu. Niektoré funkcie typické pre paketový filter budú predstavené a zmienim sa o nich v ďalších kapitolách.

4.1 Základy fungovania Netfilteru

Reťaze a politika Jedna zo skorších verzií nástroja iptables sa volala ipchains. Práve reťaze (chains) tvoria pilier fungovania paketového filtra v Linuxu, ktorú ilustruje nasledujúci obrázok [6].

Z obrázku je patrné, že tieto reťaze sú tvorené pravidlami. Paket, ktorý je zachytený v niektorej z reťaze, putuje od prvého pravidla k nasledujúcemu, pokiaľ niektorému z pravidiel nevyhovie. V prípade, že vyhovuje niektorému z pravidiel, prevedie sa niektorá z akcií, v rámci ktorej môže byť paket zahodený (DROP), odmietnuť (REJECT), prijať (ACCEPT) alebo predať inej reťaze.

Pokiaľ nevyhovie žiadnemu z pravidiel v danej reťaze, môže sa to stať v prípade základných reťazí (ref), že o jeho ďalšom osude rozhodne politika danej reťaze. K dispozícii sú dve možnosti, prvá paket prijať - ACCEPT alebo paket zahodiť - DROP. V prípade, že sa jedná o užívateľskú definovanú reťaz, je paket vrátený tam, odkiaľ bol užívateľský definovanej reťaze poslaný (RETURN).



Obr. 6: Funkcia reťaze v linuxovom paketovom filtre

4.2 Základné reťaze

Základné vstavané primárne reťaze, tvoriace základ Netfilteru, medzi ktoré patria reťaze INPUT, FORWARD a reťaz OUTPUT sú popísané na obrázku [7].

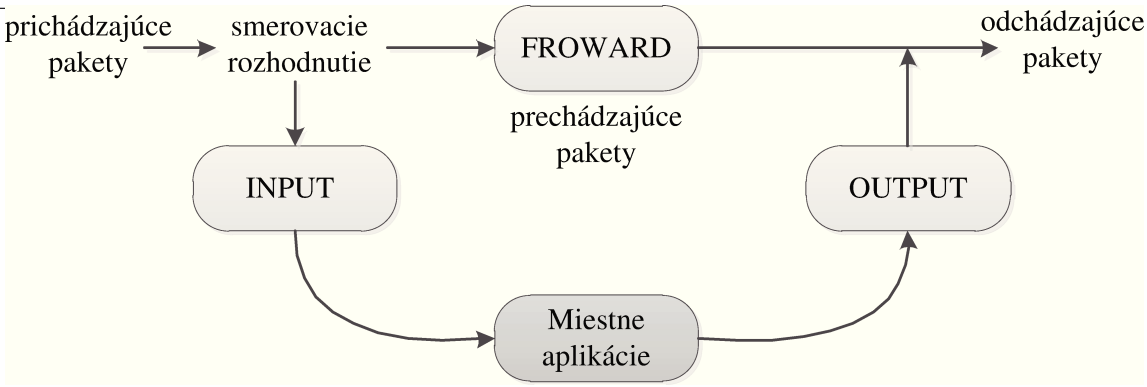
Ako náhle paket dorazí na niektoré sieťové rozhranie a je predaný cez ovládač jadru, je prevedené smerovacie rozhodnutie. Pokiaľ je paket určený lokálnemu počítaču, smeruje do reťaze INPUT. Ak je určený nejakému inému, smeruje do reťaze FORWARD. Aby bola zabezpečené predávanie paketov je nutné povoliť funkcionalitu smerovania paketov explicitne v jadre.

Okrem zmienených reťazcov INPUT, OUTPUT a FORWARD existujú ešte ďalšie dva, a to PREROUTING a POSTROUTING, ktoré sa používajú k iným účelom ako k filtrovaniu paketov. Ako je známe z názvu, PREROUTING je aktívny v dobe pred smerovaním, teda prechádzajú cez neho ako pakety určené pre lokálne zariadenia, tak i tie, ktoré sa smerujú inam. Podobne cez POSTROUTING prechádzajú pakety odchádzajúce z nášho zariadenia, rovnako ako smerované datagramy. Tieto reťazce majú zvláštny význam pri preklade adries.

4.3 Pravidlá

V jednotlivých reťaziach ide definovať pravidlá, ktoré prevádza samotné filtrovanie paketov. Tieto pravidlá majú určité podmienky, resp. kritéria, ktorým musí paket vyhovovať a príslušnú akciu, ktorá sa vykoná ak paket vyhovel pravidlu.

```
iptables -A INPUT -s 1.2.3.4 -j ACCEPT
```



Obr. 7: Schéma funkcie reťaze v linuxovom paketovom filtre

```
iptables -A INPUT -p tcp --dport 22 -j DROP
```

Tieto dva príkazy pridajú dve pravidlá na koniec reťaze INPUT. Prvé pravidlo zachytí pakety so zdrojovou adresou a prepusti ich (-j ACCEPT), t.j. príslušný paket nebude už prechádzať žiadnymi ďalšími pravidlami. Pokiaľ toto kritérium nesplnia, príde na radu druhé pravidlo, ktorému vyhovie akýkoľvek TCP paket s cieľovým portom 22 (-p tcp –dport 22), pričom tieto pakety budú zahodené (-j DROP). Pokiaľ chcem paket zahodiť, ale súčasne poslať späť ICMP správu o dôvode odmietnutia, použijem nasledujúci príkaz.

```
iptables -A INPUT -p tcp --dport 22 -j REJECT --reject-with \
  icmp-admin-prohibited
```

4.4 Prehľad základných operácií

Teraz keď sme sa už zoznámili s princípom práce s Netfilter, predstavím základné prvky pravidiel. Začnem s pridávaním a mazaním pravidiel.

```
# pridat pravidlo na koniec retaze INPUT
iptables -A INPUT ...
# pridat pravidlo na zaciatok retaze INPUT
iptables -I INPUT ...
# odobrat pravidlo – specifikovat mozete poradove cislo alebo pravidlo opisat:
iptables -D INPUT <poradove cislo pravidla>
iptables -D INPUT ...
# vymazat vsetky pravidla z retaze INPUT
iptables -F INPUT
```

Existujúce pravidlo sa odoberá z existujúceho zoznamu buď zadaním poradového čísla pravidla, alebo tak, že sa celé pravidlo opíše a namiesto voľby -A alebo -I sa zamení za -D nasledujúcim spôsobom.

```
# pridanie pravidla
iptables -A INPUT -i eth0 -s 10.0.1.5 -p tcp --dport 22 -j ACCEPT
```

```
# odobranie pravidla
iptables -D INPUT -i eth0 -s 10.0.1.5 -p tcp --dport 22 -j ACCEPT
```

Pričom zoznam všetkých pravidiel sa vypisuje nasledujúcim príkazom:

```
iptables -L -n -v
```

Voľba -L v príkaze zariadi vypísanie pravidiel, voľba -n uvádza adresy a porty číselne a voľba -v zariadi podrobný výpis.

Nasledujúci zoznam demonštruje najčastejšie používané kritéria pri tvorbe príkazov.

```
# zdrojova adresa
-s 1.2.3.4
# cielova adresa
-d 4.3.2.1
# TCP paket, cielovy port 25
-p tcp --dport 25
# UDP paket, zdrojovy port 53
-p udp --sport 53
# paket prichadzajuci z rozhrania eth0
-i eth0
# paket smeruje na rozhranie eth1
-o eth1
```

A na záver predstavím výpis základných akcií, ktoré ide s paketami prevádzať.

```
# prijat paket
-j ACCEPT
# odmietnut paket s prislusnou ICMP spravou
-j REJECT --reject-with <typ ICMP spravy>
# ticho zahodit paket
-j DROP
```

V prílohe som uviedol základny nestavový filter, ktorý je možné zostaviť na doposiaľ získaných dovednostiach.

4.5 Stavový firewall

Jedna z hlavných výhod Netfilteru je schopnosť rozlišovať, ku ktorým existujúcim spojeniam paket patrí, pokiaľ k nejakým takým vôbec patrí, a filtrovať ho na základe tejto príslušnosti. Pomocou tejto vlastnosti je možné pravidlá pre filtrovanie paketov vztiahnuť len na žiadosti o vytvorenie nového spojenia, zatiaľ čo všetky pakety patriace už vytvoreným spojením rovno prepustiť.

Stavové filtrovanie paketov zaisťuje modul state, ktorý rozširuje štyri stavy:

NEW - paket vytvára nové spojenie, alebo sa v sťahuje k spojeniu, kde doposiaľ neprebehla obojsmerná komunikácia.

ESTABLISHED - paket sa vzťahuje k prebiehajúcej obojsmernej komunikácii, teda už k vytvorenému spojeniu.

RELATED - paket vytvára nové spojenie, ale vzťahuje sa k niektorému už existujúcemu spojeniu, napr. u FTP.

INVALID - paket sa nevzťahuje k žiadnemu známemu spojeniu. Je to paket, ktorý tu nemá čo robiť, a preto je dobré ho rovno zahadzovať.

Teraz ukážem možnosť tvorby firewallu využívajúceho stavové filtrovanie:

```
1. iptables -P INPUT DROP
2. iptables -P OUTPUT ACCEPT
3. iptables -P FORWARD DROP

4. iptables -A INPUT -i lo -j ACCEPT
5. iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
6. iptables -A INPUT -m state --state INVALID -j DROP
7. iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT
8. iptables -A INPUT -p tcp --dport 80 -m state --state NEW -j ACCEPT
9  iptables -A INPUT -p icmp -j ACCEPT
10. iptables -A INPUT -j REJECT --reject-with icmp-admin-prohibited
```

Ako je jasné z predchádzajúcich dovedností (viy kap), tak prvé tri riadky špecifikujú politiku, a štvrtý riadok povoľuje všetky prichádzajúce spojenia z miestnej slučky (local loopback). Toto pravidlo je nesmierne dôležité k správnemu fungovaniu sieťovej komunikácie niektorých aplikácií, vrátane treba X serveru. Je potrebné explicitne povoliť prevádzku miestnej slučky všade tam, kde je politika pre reťaze INPUT či OUTPUT nastavená na DROP.

Piaty riadok využíva modulu state a prepúšťa všetky pakety, ktoré patria už vytvoreným spojeniam ESTABLISHED alebo novým spojeniam, ktoré k nim patria RELATED.

Šiesty riadok zahadzuje neplatné pakety INVALID, pakety, ktoré nepatria k žiadnym spojeniam. Môže sa stať, že sa zahodia pakety, o ktorých systém pre analýzu stavu connection traking alebo conntrack stratí prehľad. To sa stáva, keď dané spojenie bude ne-

aktívne dlhšie, ako je nastavený príslušný timeout. Potom conntrack dané spojenie z tabuľky vyradí, a pokaľ sa potom objavy paket patriaci k danému zahodenému spojeniu, bude považovaný za INVALID.

Pre zobrazenie obsahu a vypísanie všetkých spojení, o ktorých ma conntract prehľad slúži súbor ip_conntrack v adresári /proc/net/.

V rámci conntracku je viacej TIMEOUTov, pre vytvorené ESTABLISHED TCP spojenia. Príslušná hodnota je v súbore:

```
/proc/sys/net/netfilter/nf_conntrack_tcp_timeout_established
```

V tomto adresári sa nachádza nastavení conntracku, vrátane maximálneho počtu spojení, o ktorých si conntrack vedie záznamy - nf_conntrack_max.

Riadky 7 a 8 povoľujú nové TCP spojenia smerujúce na porty 22 pre SSH a 80 pre HTTP.

Na deviatom riadku sa povoľujú ICMP pakety, aj keď to nie je nutné a server bude fungovať a bude prístupný aj bez toho, ale hodí sa povoliť minimálne ping, teda ICMP echo request.

Posledné pravidlo zariadi, že paket, ktorý nevyhovie žiadnym predchádzajúcim pravidlám, bude odmietnutý, teda zahodený, ale odosielateľovi sa vráti ICMP správa o nedoručení. Keby tam nebolo toto pravidlo, paket by bol taktiež zahodený, pretože politika reťaze INPUT je nastavená na DROP.

4.6 IPv6 a Netfilter

V dnešnej dobe, keď rýchlo dochádzajú adresné priestory IPv4 je neodvratný prechod na IPv6. Netfilter je na tento pomalý prechod už dlhšie pripravený, pričom s tým dôležitým faktom, že filtre pre IPv4 a IPv6 prevádzku sú oddelené. Preto je nutné nastaviť filtrovanie služieb, ktoré využívajú IPv6 pomocou nástroja ip6tables, ktorého syntax je totožná s nástrojom iptables.

U väčšiny pravidiel je možné ponechať úplne rovnaké argumenty, iba niekde je nutné zohľadniť rozdiely medzi protokolmi IPv4 a IPv6. Ekvivalentom firewallu z predchádzajúcej kapitoly (kap) by vypadal po menšej úprave totožne až na rozdiely v argumentoch na piatom a desiatom riadku.

```
5. ip6tables -A INPUT -p ipv6-icmp -j ACCEPT
.
.
.
10. ip6tables -A INPUT -j REJECT --reject-with icmp6-adm-prohibited
```

4.7 Ako integrovať firewall do systému

Väčšina Linuxových distribúcii nemá v štartovacích skriptách zohľadnené nastavovanie firewallu, takže neostáva nič iné ako si prípadný firewall napísať ako shelový skript[6] a začleniť ho do bootovacieho procesu */etc/network/if-up.d/*, alebo že príslušný skript umiestnim ako pre-up skript v konfiguračnom súbore pre sieťové rozhranie, ktorým je */etc/network/interfaces* následovne:

```
iface eth0 inet static
      address 10.0.0.254
      netmask 255.255.255.0
      gateway 10.0.0.138
      pre-up /root/firewall.sh
```

4.8 Ako integrovať firewall do systému

Netfilter umožňuje definovať vlastné reťaze, a tým pravidla paketového filtru logický členiť a zjednodušiť. Základné reťaze INPUT, OUTPUT a FORWARD nie je možné - zmazať, iba vyčistiť. Je možné vytvoriť novú reťaz a už v existujúcich sa na ňu potom odkazovať. Pošleme daný paket, ktorý vyhovie danému pravidlu do danej reťaze. Užívateľom definované reťaze nemajú politiku, takže pokiaľ paket užívateľským pravidlom prejde bez toho, aby bz bol zachytený nejakým pravidlom, vracia sa späť do reťaze, odkiaľ bol do užívateľský definovanej reťaze odoslaný.

Najlepšie to bude demonštrovať na našom firewalle, ktorý rozšírim o užívateľské reťaze.

```
1. iptables -F
2. iptables -X
3. iptables -P INPUT DROP
4. iptables -P OUTPUT ACCEPT
5. iptables -P FORWARD DROP

6. iptables -N ssh
7. iptables -A ssh -s 10.0.0.0/8 -j ACCEPT
8. iptables -A ssh -s 43.21.12.34 -j DROP
9. iptables -A ssh -m limit --limit 5/sec --limit-burst 100 -j ACCEPT
10. iptables -A ssh -j DROP

11. iptables -A INPUT -i lo -j ACCEPT
12. iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
13. iptables -A INPUT -m state --state INVALID -j DROP
14. iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j ssh
15. iptables -A INPUT -p tcp --dport 80 -m state --state NEW -j ACCEPT
16. iptables -A INPUT -p icmp -j ACCEPT
17. iptables -A INPUT -j DROP
```

[6] Interpret pre príkazový priadok v unixových distribúciách.

Prvý riadok sa postará o to, že odstráni všetky pravidlá zo všetkých reťazí, druhý príkaz odstráni všetky prázdne užívateľom definované reťaze. Na riadku 7 a 8 nasleduje vytvorenie novej reťaze s názvom ssh, kde je načítaný whitelist a blacklist. riadok 9 zabezpečí obmedzenie počtu spojení na SSH. Posledné pravidlo v reťazi zahadzuje všetky pakety, ktoré neodpovedajú niektorému z pravidiel v reťazy.

Na reťaz ssh je odkázané na riadku 14, kde je užívateľský definovaný reťazec predaný parametru -j, ktorý určuje, čo sa ma z daným paetom vyhovujúcemu danému pravidlu vykonať. V mojom prípade paket najprv dorazí do reťaze INPUT na riadku 11. Pakety, ktoré vyhovia pravidlu na 14 riadku, budú poslané do reťaze ssh.

4.9 Logovanie paketov

V prípade, ak chceme špecifickú sieťovú prevádzku monitorovať o niečo bližšie, je možné jednotlivé pakety, ktoré budú vyhovovať pravidlu, nechať logovať napríklad takto:

```
   [...]

1. iptables -N
2. iptables -A strange -s 10.0.0.0/8 -j LOG \
     --log-prefix "iptables-strange-local:␣"
3. iptables -A strange -s 12.23.34.45 -j LOG \
     --log-prefix "iptables-strange-offender:␣"
4. iptables -A strange -j DROP

   [...]

5  iptables -A INPUT -m state --state INVALID -j strange

   [...]
```

Všetky pakety v stave INVALID sú poslané do užívateľskej definovanej reťaze `strange`, v ktorej dochádza k výberu jednej siete a jednej IP adresy, u ktorej sa bude prevádzať logovanie. Všetko ostatné sa potom zahadzuje.

Pokiaľ paket vyhovie pravidlu s cieľom LOG, je zalogovaný, ale pokračuje ďalej. preto na riadku 4 sa definovalo pravidlo, ktoré všetky pakety v tejto reťazi zahodí. Samotné logovanie ma nepovinnú voľbu –log-prefix, ktorá umožňuje špecifikovať, čím bude hlásenie na riadku so záznamom v logu načítať.

4.10 Reťaz FORWARD

GNU/Linux môže fungovať ako smerovač, teda umožňuje predávať pakety z jedného rozhrania na iné, z jednej siete do druhej. Pakety, ktoré nie sú určené pre daný počítač, teda ním len prechádzajú, putujú v rámci paketového filtra reťazou `FORWARD`. Preto som v predchádzajúcich príkladoch tuto politiku pre reťaz `FORWARD` nastavoval na `DROP`.

Ak chceme firewall využívať aj ako smerovač, je potrebné smerovanie povoliť v konfigurácií jadra v *ip_forward* nastavením premennej forwarding na 1, ktoré sa avšak dočasné - jednorázové.

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

Trvalé povolenie smerovania je možné editáciou súboru */etc/sysctl.conf* a nastavenie premennej *forwarding* na hodnotu jedna.

```
net.ipv4.conf.default.forwarding = 1
```

Aj tu je potrebné rozoznávať protokol IPv4 a IPv6 a nastavovať predávanie paketov individuálne.

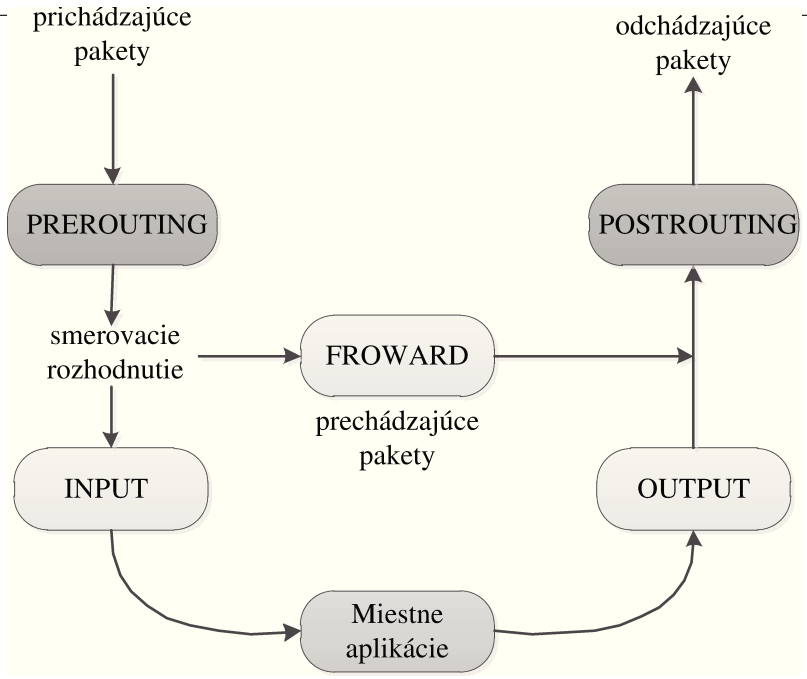
Samotné filtrovanie predávaných paketov môže vypadať nasledovne.

```
1. iptables -P FORWARD DROP
2. iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
3. iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT
4. iptables -A FORWARD -s 10.0.0.0/24 -d 10.0.1.0/24 -j ACCEPT
5. iptables -A FORWARD -s 10.0.1.0/24 -d 10.0.0.0/24 -j ACCEPT
```

Nastaví vychodzia politika reťaze FORWARD, potom sa povolí priechod paketov, ktoré odpovedajú už vytvoreným spojeniam alebo sa ich tykajú a na treťom riadku je povolený celá sieťová prevádzka smerujúca z rozhrania eth0 na rozhranie eth1. Potom na posledných dvoch riadkoch je povolená prevádzka v oboch smeroch medzi sieťami `10.0.0.0/24` a `10.0.1.0/24`.

4.11 Tabuľky v Netfilteru

Paketový filter v Linuxe, ako je to jasné z názvu, pracuje s tabuľkami. Preto pokiaľ v rámci použitia nástroja iptables nezadáte názov tabuľky, s ktorou chcete pracovať, využije sa vychodzia tabuľka filter, v ktorej prebieha samotné filtrovanie paketov. Existujú ďalšie tri tabuľky, nat, mangle a raw, kde v každej z týchto tabuliek sú rôzne vychodzie reťaze, s ktorými je možné pracovať. Tabuľka mangle slúži k modifikovaní alebo značkovaní paketov, tabuľka nat pre preklad adries a tabuľka raw sa používa pre označenie paketov, ktoré sa majú vyhnúť sledovaniu spojenia (connection tracking).



Obr. 8: Prechádzanie paketov linuxovým paketovým filtrom.

Tabuľkami mangle a raw sa nebudem zaoberať, nakoľko priamo nesúvisia s paketovým filtrom. Preto svoju pozornosť upriem na tabuľku nat a dve reťaze v nej, ktoré nám rozšíria schému prechádzania paketov Netfilterom.

Ako je možné vidieť na obrázku [8], tak ďalšie dve reťaze PREROUTING a POSTROUTING, ktoré patria tabuľke nat sú vyznačené na zeleno. Tieto dve reťaze zachytávajú pakety ako tesne pred smerovacím rozhodnutím PREROUTING, tak aj tesne pred opustením počítača POSTROUTING.

4.12 Netfilter a NAT

V NAT (Network address translation) existujú dve formy, ako zdrojový NAT (SNAT), kde sa u paketov mení zdrojová adresa, a cieľový NAT (DNAT), v rámci ktorého dochádza k úprave cieľovej adresy. Špeciálnym prípadom je maškaráda, ktorá je vhodná pre pripojenie s dynamickou IP adresou.

4.13 Presmerovanie portov a DNAT

Presmerovanie portov sa využíva v mnoho oblastiach, keď chceme presmerovať niektoré porty na server v domacej sieti s jednou verejnou IP adresou.

Na nasledujúcom príklade ukážem presmerovanie portov v praxy.

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j DNAT \
  --to 10.0.2.200:8080
iptables -A FORWARD -i eth1 -o eth2 -d 10.0.2.200 -p tcp --dport 8080 \
  -j ACCEPT
```

Na prvom riadku je pravidlo, ktoré zmení cieľovú adresu a cieľový port paketov smerujúcich na port 80 rozhrania eth1 na IP adresu 10.0.2.200 a port 8080. Toto pravidlo iba zmení cieľovú adresu paketu, ktorý pravidlu vyhovuje a nič iného s paketom nerobí. Takýto paket prejde smerovacím rozhodnutím a putuje do reťaze FORWARD, kde je treba zistiť priechod, riadok 2 ho pošle na rozhranie eth2, pričom smerovanie paketov musí byť povolené.

V rámci DNAT je možné presmerovať všetku sieťovú prevádzku z jedného rozhrania na iné rozhranie.

4.14 Maškaráda a SNAT

Maškaráda sa najčastejšie používa v prípade, ak máme od poskytovateľa internetu pridelenú dynamicky IP adresu napríklad na rozhraní eth0, a my chceme za tento počítač schovať celú vnútornú sieť. Ta zaručí, že sa u všetkých odchádzajúcich paketov smerujúcich na dané rozhranie zmení zdrojová IP adresa na IP adresu pridelenú danému rozhraniu.

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Základným rozdielom medzi maškarádu a SNAT v rámci iptables je, že pri maškaráde je absencia nutnosti špecifikovať zdrojovú adresu, jednoducho sa použije IP adresa výstupného rozhrania. Avšak pri zhodení daného rozhrania Netfilter "zabudneïnformácie o všetkých relevantných spojeniach. Ak sa táto adresa nemení, tak namiesto maškarády je vhodné použiť klasický SNAT.

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 10.0.5.15
```

4.15 Niektoré ďalšie moduly Netfilteru

Netfilter ponúka celú radu modulov, ktoré umožňujú filtrovať pakety podľa doplňujúcich kritérií.

4.15.1 Vlastník paketu - owner

Niekedy nie je od veci monitorovať na serveru iba prichádzajúcu komunikáciu, ale aj odchodziu. Pri tejto odchádzajúcej komunikácii sa môže hodiť filtrovať pakety na základe UID alebo GID procesu.

```
iptables -A OUTPUT -o eth0 -m owner --uid-owner podozrivy-uzivatel \
-m limit --limit 1/s --limit-burst 10 -j LOG
```

Toto pravidlo loguje odchádzajúce pakety generované procesmi užívateľa podozrivy-uzivatel smerujúci na sieťové rozhranie eth0. Modul limit zabezpečí ochranu proti preplneniu logov pri generovanom príliš veľkej prevádzke.

4.15.2 Počet spojení - connlimit

Pokiaľ potrebujeme obmedziť určitú prevádzku po prekonaní určitých spojení využijeme modul connlimit.

```
iptables -A FORWARD -i eth2 -o eth0 -m connlimit --connlimit-above 10 \
  -j REJECT
```

4.15.3 Multiport

V prípade, ak je potrebné v jednom pravidle definovať viacej TCP/UDP portov, využívame modul multiport. Špecifikovanie niekoľkých portov za sebou oddeľujeme čiarkou, a ak chceme špecifikovať rozsah portu využíva sa na určenie rozsahu dvojbodka. Takto je možné špecifikovať až 15 portov, pričom rozsah je počítaný ako dva porty [9].

```
iptables -A INPUT -p tcp -m multiport --ports 22,80,40000:50000 -j ACCEPT
```

5 Návrh vlastného riešenia systému s IPS

5.1 Zapojenie

Cieľom diplomovej práce bolo navrhnúť a implementovať počítačovú sieť so systémami zo zameraním sa na bezpečnosť tejto siete proti rôznym nežiadaným prienikom z vonkajšej siete, neštandardného chovania sa v tejto sieti. Na obrázku [9] sú zachytené jednotlivé časti siete a sieťové prvky v nej obsiahnuté.

Ako je patrné z obrázka, tak je sieť rozdelená na niekoľko hlavných zón siete - podsieti a centrálneho spojovacieho prvku medzi nimi.

5.1.1 Demilitarizovaná sieť

Demilitarizovaná zóna (angl. *Demilitarized Zone*, **DMZ**) je od ostatnej privátnej siete oddelená a tak sama o sebe tvorí vlastnú sieť. Nachádzajú sa tu tri servery, ktoré sprostredkovávajú služby medzi vnútornou a vonkajšou sieťou.

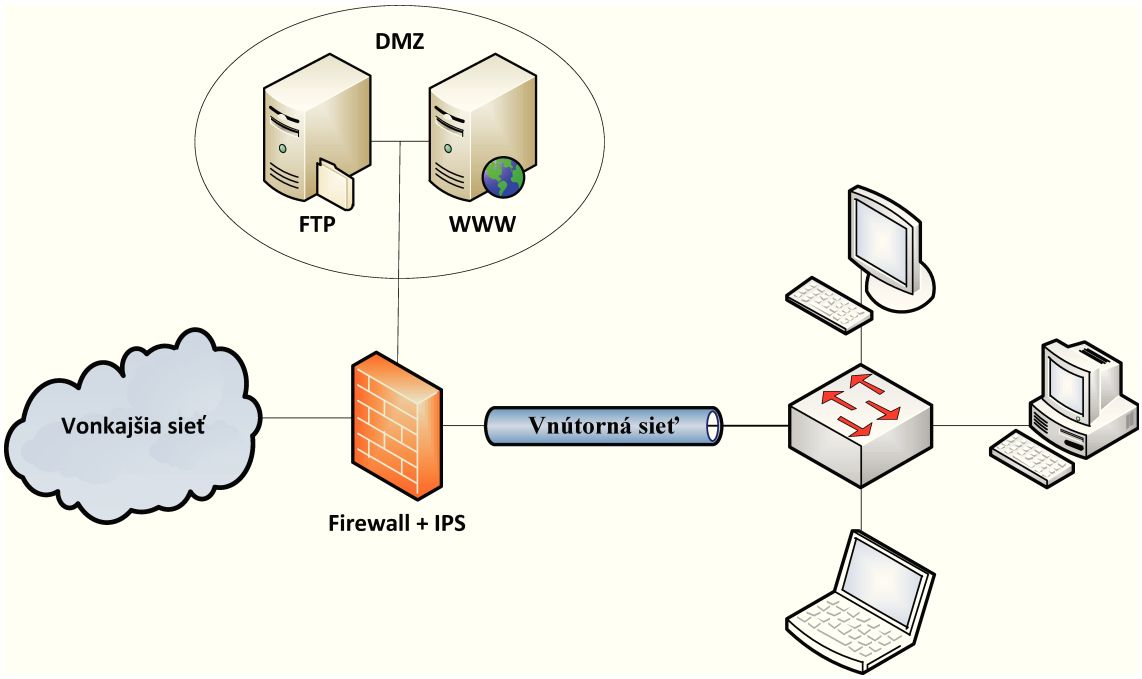
V DMZ sú inštalované tri servery pre prístup na webové stránky, zdieľanie súborov a databázu, postavené na platforme Linux distribúcie Ubuntu Server 11.10. Servery majú rozdielnu úroveň prístupu pre vonkajšiu a vnútornú sieť, väčšinou užívatelia z vnútornej siete majú väčšie právomoci. K prvkom DMZ nie je možné pristupovať priamo, pretože sú skryté. Užívateľovi sú k dispozícii pod IP adresou vnútorného či vonkajšieho rozhrania IPS.

5.1.2 Firewall + IPS

Významným prvkom siete je Firewall s IPS, ktorý slúži ako centrálny prepojovací uzol medzi demilitarizovanou zónou, vnútornou a vonkajšou sieťou. Vykonáva funkciu firewallu, smerovača a DHCP servera, ktorého beh zaisťuje operačný systém *Ubuntu Server 11.10*. Vďaka centrálnemu umiestneniu je výhodné nasadenie systému prevencie proti narušeniu, realizované open-source aplikáciou *Snort*.

Pri voľbe programového vybavenia jednotlivých prvkov siete bol braný zreteľ na niekoľko cieľov ako cena, účinnosť, bezpečnosť či náročnosť riešenia. Preto bola celá sieť, ktorá je na obrázku [9], realizovaná vo virtuálnom prostredí, z dôvodu časového deficitu.

V nasledujúcich kapitolách sa budem venovať hlavne popisu jednotlivých krokov pri realizácií a správnej konfigurácií prvkov siete. Správna funkcia bude overená prevedením niekoľkých útokov pomocou penetračných nástrojov ako sú *Nmap*, *hping*, ...



Obr. 9: Schéma siete proti narušeniu

5.2 Inštalácia a konfigurácia

V tejto kapitole sa budem zaoberať predovšetkým správnej konfigurácií prvkov siete, inštalácií služieb na servery a prvky a stavbou stavového paketového filtru. Pre sieť na obrázku [9] sa definovali rozsahy adries pre jednotlivé podsiete, dbajúc na prehľadnosť boli zvolené rozsahy:

- 10.1.1.0/24 - Vnútorná sieť
- 10.2.2.0/24 - Demilitarizovaná sieť
- 192.168.0.0/24 - Vonkajšia sieť

5.2.1 Vnútorná sieť

Užívatelia, ktorí budú umiestnený vo vnútornej sieti budú mať dynamicky pridelené IP adresy pomocou DHCP serveru umiestneného na centrálnom prvku. Rozsah prideľovaných adries je 10.1.1.10 až 10.1.1.200.

5.2.2 DMZ

Hlavnými servermi demilitarizovanej zóny, ktoré sa budú využívať sú web server, SFTP server (angl. *SSH File Transfer Protocol* a MySQL databáza. V rámci úspor hardwarových prostriedkov sú tieto dva servery umiestnené na jednom počítači. Tento počítač je vybavený operačným systémom *Ubuntu Server 11.10* a len jedným sieťovým rozhraním. Ro-

zhraniu sa staticky nastavila IP adresa *10.2.2.23/24*, cez ktoré sa budú poskytovať zmienené služby na portoch:

- 22 - **SSH** (angl. *Secure Shell*)
- 80 - **HTTP** (angl. *Hypertext Transfer Protocol*)
- 443 - **HTTPS** (angl. *HTTP Secure*)
- 2222 - **SFTP** (angl. *SSH File Transfer Protocol*)
- 3306 - **MySQL** (angl. *My Structured Query Language*)

WWW server Pre uživateľov ako vnútornej, tak aj vonkajšej siete boli zriadené webové stránky, ktoré sa dajú zobrazovať cez šifrovaný protokol HTTPS. Ako overený a vhodný kandidát bol zvolený open-source program *Apache2*.

Inštaláciu webového servera *Apache2* a komponenty *SSH* realizujúcej šifrovanie spojenia zabezpečili príkazy:

```
sudo apt-get install apache2
sudo a2enmod ssl
```

Šifrované spojenie je ešte potrebné povoliť, a to nasledovne:

```
sudo /etc/init.d/apache2 force-reload
```

K realizácií šifrovaného spojenia medzi klientom a serverom som použil vlastný certifikát podpísaný vlastným kľúčom dĺžky 1024 bitov.

```
cd /etc/apache2
sudo openssl genrsa -des3 -out server.key 1024
```

Vytvorenie firemného certifikátu pre webový server vyžadovalo vyplniť dotazník a údaje, ako napríklad názov organizácie alebo e-mailová adresa. Vytvorenie certifikátu spoločne s vyplnenými údajmi sú nižšie.

```
sudo openssl req -new -key server.key -out server.csr

Country Name (2 letter code) [AU]:CZ
State or Province Name (full name) [Some-State]:Czech
Locality Name (eg, city) []:Ostrava
Organization Name (eg, company) [Internet Widgits Pty Ltd]:VSB-TU
Organizational Unit Name (eg, section) []:FEI
Common Name (eg, YOUR name) []:Michal Krupa
Email Address []:michal.krupa@vsb.cz
```

Podpísanie autorizačného certifikátu vygenerovaným súkromným kľúčom na dobu 1 rok, teda 365 dní sa realizovalo následujúco.

```
sudo openssl x509 -req -days 365 -in server.csr -signkey server.key \
-out server.crt
```

Bolo nutné správne umiestniť vygenerovaný kľuč a certifikát do určených zložiek.

```
sudo cp server.crt /etc/ssl/certs/
sudo cp server.key /etc/ssl/private/
```

Ďalej sa musel editovať konfiguračný súbor pre zabezpečené spojenie v adresári *cd /etc/apache2/sites-available*.

```
sudo nano default-ssl
```

V tomto konfiguračnom súbore pre zabezpečené spojenie sa odkomentovali riadky a zmenili sa cesty k certifikáte a súkromnému kľúču [10].

```
SSLEngine on
SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
SSLCertificateFile /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
```

K správnemu behu webového serveru bolo ešte potrebné povoliť základné SSL a reštartovať Apache2.

```
sudo a2ensite default-ssl
sudo /etc/init.d/apache2 restart
```

SFTP server V sieti DMZ sa nachádza ďalší server a to server na zdieľanie dát, na ktorý majú prístup ako užívatelia z vnútornej siete, tak aj z vonkajšej. Pretože nasedenie FTP nie je vhodnou voľbou, pretože prihlasovacie údaje i prenášané dáta sú v nešifrovanej forme, rozhodol som sa preto pre implementáciu Secure FTP pomocou open-source aplikácie *proftpd*, ktorá sa inštalovala nasledovne:

```
sudo apt-get install proftpd
```

Všetky potrebné požadované parametre pre nastavenie správneho chodu sa nachádzajú v konfiguračnom súbore v adresári */etc/proftpd/proftpd.conf*.

K správnemu bezpečnému nastaveniu servera bolo potrebné nastaviť niekoľko bodov. Medzi hlavné požiadavky je možné zaradiť podporu šifrovanej komunikácie, prihlásenie sa viacero užívateľov, ktorý majú možnosť po prihlásení sa iba sťahovať súbory

z adresára */home/ftp/download*. Výnimku tvorí adresár */home/ftp/upload*, kde je možný aj zápis.

K realizácií šifrovaného spojenia som postupoval obdobne ako pri generovaní kľúča a vytvorení certifikátu u web servera [5.2.2]. Navyše sa generoval CA certifikát [11].

```
sudo openssl genrsa -des3 -out ca.key 1024
sudo openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

Pre správny beh SFTP servera sa musel editovať konfiguračný súbor a doplniť nasledujúce riadky, aby sa lokalizoval kľúč a certifikáty.

```
. . .

# certifikaty servera
TLSRSACertificateFile /etc/ftpcert/server.crt
TLSRSACertificateKeyFile /etc/ftpcert/server.key

# CA certifikacna autorita
TLSCACertificateFile /etc/ftpcert/ca.crt

. . .
```

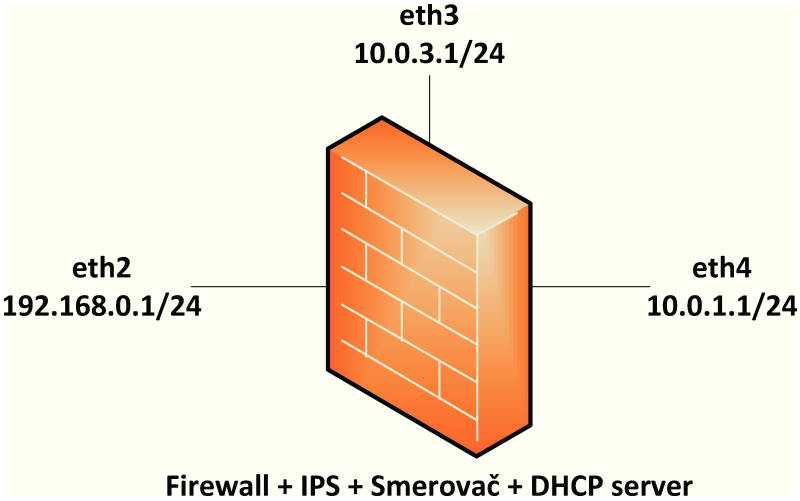
Taktiež je možné určovať privilégia užívateľov a skupín k adresárom súborového servera. Napríklad povoliť užívateľom skupiny *ftpusers* sťahovať a nahrávať súbory. Je nutné dodať, že port som zmenil z 21 na 2222, aby nedošlo k použitiu šifrovanej voľby ku konfliktu s SSH, ktorý je dostupný na TCP porte 22 [12].

MySQL Hlavnou podstatou nástroja Snort je logovanie a ukladanie generovaných výstrah do logu, v sofistikovanejšom prevedení do databáze. táto vlastnosť Snortu prináša nové možnosti, pretože uložené výstrahy môžu poslúžiť pre rôzne štatistické metódy. Preto som sa rozhodol pre databázovú open-source službu MySQL, ktorá je dostupná na TCP porte 3306.

5.2.3 Firewall s IPS

Tento centrálny prvok je vďaka svojmu umiestneniu najvýznamnejším z celej časti siete. Vykonáva dôležité funkcie. Vykonáva funkciu stavového filtra paketov, smerovača, servera pre dynamické prideľovanie adries a kontrolóra prichádzajúcej i odchádzajúcej komunikácie. Disponuje troma rozhraniami, ktorých IP adresy sú prideľované staticky. Firewall spolu s IPS beží na overenom operačnom systéme *Ubuntu Server 11.10*.Nastavenie sieťových rozhraní je zachytené na obrázku [10].

DHCP V privátnej sieti sú zariadenia bez pridelených IP adries, preto bolo potrebné na centrálnom prvku zariadiť beh servera pre dynamické prideľovanie adries. Inštalácia



Obr. 10: Nastavenie sieťových rozhraní na Firewalle s IPS

servera na operačnom systéme *Ubuntu Server 11.10* sa vykoná:

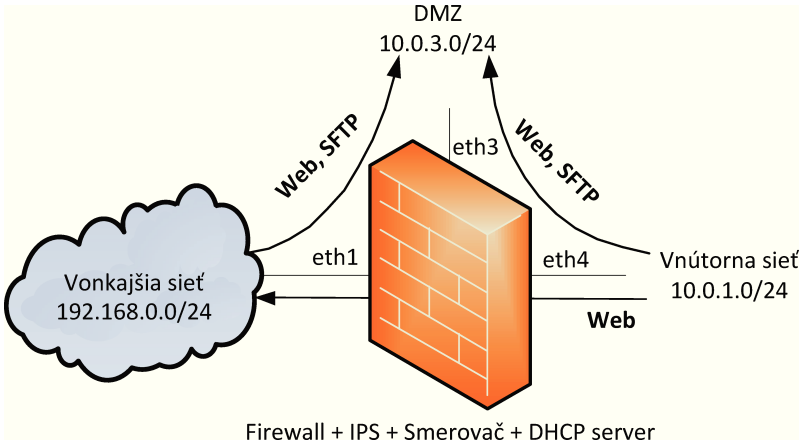
```
sudo apt-get install dhcp3-server
```

V konfiguračnom súbore */etc/dhcp3/dhcpd.conf*, sa editovali riadky pre správnu funkčnosť v tejto sieti.

```
subnet 10.0.1.0 netmask 255.255.255.224 {
  range 10.0.1.20 10.0.1.200;
  option domain-name-servers 192.168.0.2;
  option domain-name 10.0.1.1;
  option broadcast-address 255.255.255.0;
  option routers 10.0.1.1;
  default-lease-time 600;
  max-lease-time 7200;
}
```

Zariadenia v privátnej časti siete nastavili sieťové rozhrania tak, aby prijímali od DHCP servera IP adresy v rozsahu od `10.0.1.20` až do `10.0.1.200` s potrebnými parametrami ako vychodzia brána, adresa DNS servera a inými voliteľnými.

```
auto eth1
iface eth1 inet dhcp
```



Obr. 11: Dostupnosť poskytovaných služieb

5.3 Iptables

Ako vhodný nástroj pre tvorbu stavového paketového filtru som zvolil Iptables. Na zostrojenie vhodného firewallu som dbal na niekoľko hlavných vlastností, ktoré by môj firewall umiestnený medzi vnútornou, vonkajšou sieťou a DMZ mal mať:

1. Vnútorná sieť nesmie byť vidieť z vonkajšej.
2. Z vonkajšej siete je dostupná DMZ na portoch 22, 80, 443 cez ip adresu vonkajšieho rozhrania firewallu.
3. Na servery v DMZ je možné pristupovať z vnútornej siete iba cez protokoly HTTP, HTTPS, SFTP, SSH.
4. Z vnútornej sieti je možné pristupovať na web vo vonkajšej sieti.

Dostupné služby v sieti sú zachytené na obrázku [11].

Všetky pravidlá `iptables` som uložil do shell súboru *iptables* a integroval do bootovacieho procesu, kde sa bude spúšťať po štarte systému [4.7]. Tento shell súbor je priložený ako príloha na záznamovom médiu, ktoré je súčasťou diplomovej práce.

5.3.1 Smerovač

Smerovač (angl. *Router*) umiestnený v centru medzi sieťami zabezpečuje správne smerovanie paketov, aj keď DMZ nie je vidieť z vonkajšej siete. Užívateľom sú k dispozícii odoslaním požiadavku na vonkajšie rozhranie. Taktiež zabezpečuje smerovanie medzi vnútornou sieťou a DMZ a vonkajšou sieťou.

Prvým krokom pri sprístupnení služieb medzi jednotlivými sieťami bolo povoliť predávanie komunikácie medzi jednotlivými rozhraniami v jadre systému [4.10], ja osobne

som sa priklonil k prvému spôsobu.

Bolo potrebné nastaviť smerovanie z vonkajšej siete do DMZ na portoch 80 pre HTTP, 443 pre HTTPS a 2222 pre SFTP. Tieto vyhovujúce pakety sa zaradia do fronty, z ktorej ich vyzdvihne služba Snort, ako je to na obrázku [5] a porovnáva ich s vlastnými pravidlami [2.5].

```
# Bad TCP pakety su neziaduce

$IPTABLES -A FORWARD -p tcp --tcp-flags SYN,ACK SYN,ACK -m state \
--state NEW -j REJECT --reject-with tcp-reset

# Pravidla pre LAN

$IPTABLES -A FORWARD -p ICMP -i $lan_if -j ACCEPT
$IPTABLES -A FORWARD -p TCP -i $lan_if -o $dmz_if --dport 80 -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Pravidla pre WAN

$IPTABLES -A FORWARD -p TCP -i $wan_if -d $apache --dport 80 -j ACCEPT

# Pravidla pre DMZ

$IPTABLES -A FORWARD -p ALL -i $dmz_if -o $wan_if -j ACCEPT

# Snort

$IPTABLES -A FORWARD -i $wan_if -m state --state NEW,RELATED, \
ESTABLISHED -d $apache -p tcp -j QUEUE
$IPTABLES -A FORWARD -i $dmz_if -o eth2 -m state --state RELATED, \
ESTABLISHED -p tcp -j QUEUE
```

5.3.2 Preklad adries a portov

Nastavenie tabuliek PREROUTING v NetFilteru a editáciu požiadavkou [4.14], prichádzajúcich na vonkajšie rozhranie určené pre DMZ, pre službu HTTP, HTTPS a SFTP, kde sa modifikuje cieľová IP adresa cieľový port môže vypadať následovne:

```
# Pravidla pre WAN

$IPTABLES -t nat -A POSTROUTING -o $wan_if -j SNAT --to $wan_ip
$IPTABLES -t nat -A PREROUTING -p TCP -i $wan_if --dport 80 \
-j DNAT --to-destination $apache
$IPTABLES -t nat -A PREROUTING -p TCP -i $wan_if --dport 443 \
-j DNAT --to-destination $apache
$IPTABLES -t nat -A PREROUTING -p UDP -i $wan_if --dport 22 \
-j DNAT --to-destination $sftp:2222
```

Prvé pravidlo je veľmi dôležité s použitím reťaze POSTROUTING, pretože celú vnútornú sieť pri komunikácii s vonkajšou, ukrýva za vonkajšiu ip adresu, ktorou je možné vystupovať vo verejnej sieti.

5.4 Snort

Inštalácia open-source nástroja pre detekciu narušenia Snort nemôže byť prevedená z konzolového riadku, ale musí byť prevedená inštaláciou zo zdrojových kódov. Aby Snort pracoval v režime inline, teda spolupracoval s iptables, je pred samotnou inštaláciou Snortu potrebné nainštalovať nástroje.

```
sudo apt-get install bison flex gcc g++ zlib1g-dev autoconf byacc
```

Po úspešnom doplnení potrebných nástrojov, ktoré sú bezprostredne potrebné k ďalším inštaláciám, je nutné stiahnuť knižnice, dôležité pre spustenie a prevádzkovanie nástroja Snort v režime IPS.

```
pcre-8.30.tar.gz
libpcap-1.2.1.tar.gz
libdnet-1.11.tar.gz
daq-0.5.tar.gz
```

5.4.1 libpcap

Knižnica slúžiaca na odchytávanie sieťovej komunikácie cez dané rozhranie, aj takej, ktoré nie sú adresované danému zariadeniu[7]. Dokáže do tejto siete vysielať vytvorené, resp. odchytené pakety. Vypisuje zoznam sieťových rozhraní.

Libpcap dokáže verne zachytávať a ukladať sledovanú sieťovú komunikáciu od úrovne spojovej vrstvy a neskôr sa k nej vrátiť za účelom jej ďalšej analýzy [13].

Balíček libpcap som stiahol z domovských stránok [14] a po rozbalení ho nainštaloval nasledovne.

```
wget http://www.tcpdump.org/release/libpcap-1.2.1.tar.gz
sudo tar zxvf libpcap-1.2.1.tar.gz
cd libpcap-1.2.1/
sudo ./configure
sudo make
sudo make install
```

[7] Sieťová karta musí byť v promiskuitnom režime

5.4.2 libdnet

Libdnet poskytuje zjednodušené rozhranie pre prácu s paketmi na nízkoúrovňových hladinách, vrátane manipulácie s:

- adresou siete
- arp cache a smerovacou tabuľkou
- firewallom
- IP tunelovaním
- IP paketom a eternetovým rámcom

Pre prístup a nastavenie pravidiel firewallu na lokálnom systéme je definovaný v fw-*.c.

Balík sa nainštaluje ako bolo tomu aj pri Libpcap [15].

```
wget http://prdownloads.sourceforge.net/libdnet/libdnet-1.11.tar.gz
sudo tar zxvf libdnet-1.11.tar.gz
cd libdnet-1.11/
sudo ./configure
sudo make
sudo make install
```

5.4.3 libnet

Tento softwarový nástroj umožňuje zostavovanie paketov, pričom Snort vyžaduje verziu balíku 1.0.x, ktorá ale na domovských stránkach nie je dostupná, preto balík bol stiahnutý z [16], rozbalenie a inštaláciu som vykonal nasledovne:

```
wget http://ips-builder.googlecode.com/files/libnet-1.0.2a.tar.gz
sudo chown -R misho:misho Libnet-1.0.2a
sudo tar zxvf Libnet-1.0.2a.tar.gz
cd libnet-1.0.2a/
sudo ./configure
sudo make
sudo make install
```

DAQ a pcre Tieto balíky boli stiahnuté pre daq vo verzií daq-0.6.2.tar.gz z [2] a pre pcre verzie pcre-8.30.tar.gz zo stránok [17]. Rozbalenie balíkov a ich inštalácia prebiehla štandardným spôsobom ako to bolo u balíkoch vyžšie zmienených.

5.4.4 MySQL

Databáza bude slúžiť pre ukladanie výstrah pri prevenci proti narušeniu. Je to vhodnejšie, a čo sa týka využitia, sofistikovanejší spôsob záznamenavania výstrah. K uloženým výstraham je možne, vďaka spolupráci s webovým serverom prietúpovať pohodlne cez webové rozhranie. Dáta uložené na tejto databáze sú vhodné ako vstup pre rôzne štatistické metódy, pri ktorých môžme odkryť užitočné informácie, ktoré môžu byť v budúcnosti nápomocné k lepšej ochrane siete.

Inštaluje sa balíček `mysql-client`, `libmysqlclient-dev` a `mysql-server` a potom je nutné vytvoriť tabuľku zviazanú s nástrojom Snort.

```
mysql -u root -p
grant ALL PRIVILEGES on snort.* to snort@10.0.2.23 with GRANT option;
grant ALL on snort.* to snorby@10.0.2.23;
SET PASSWORD FOR snort@10.0.2.23=PASSWORD('snortPASSWD');
SET PASSWORD FOR snorby@10.0.2.23=PASSWORD('snorbyPASSWD');
```

Prácu uľahči, prekopírovanie schémy databázy create_mysql z adresáru Snortu, ktorá obsahuje všetky potrbné atríbuty.

5.4.5 Barnyard2

Barnyard je výstupný systém pre Snort. Snort vytvára špeciálny binárny výstupný formát volaný unified. Barnyard tento súbor číta a potom ho posiela na koncovú databázu. Riadi odosielanie udalosti do databázy a ukladá ich, keď databáza dočasne nemôže prijať spojenie.Tak odbremeňuje Snort.

V prípade ak je potrebné výstrahy ukladať do databázy a pristupovať k ním, je potrebné nainštalovať systém spolupracujúci so Snortom a to Barnyard2 dostupný zo stránok [18].

```
tar zxvf barnyard2-1.9.tar.gz
cd barnyard2-1.9
sudo ./configure --with-mysql
sudo make && make install
```

Ďalej je nutné pre správnu komunikáciu s databízou nastaviť v konfiguračnom súbore `barnyard2.conf` parametre.

```
output database: log, mysql, user=snort password=snort \
  dbname=snort host=10.0.2.23
```

5.4.6 Snorby

Výstrahy je možné zobraziť aj v grafickej forme cez webové rozhranie a to pomocou veľmi šikovného nástroja Snorby [19].

Obr. 12: Snorby

Tento nástroj požaduje podporu *Ruby, Rails* a *ImageMagick*. Samotný balík Snorby stiahnem z domovských stránok a rozbalím ho v hlavnom adresáry pre webové služby.

Je nutné vytvoriť v databáze novú tabuľku určenú pre Snorby a nastaviť správne konfiguráciu `database.yml` pre pripojenie sa k databáze.

```
snorby: &snorby
  adapter: mysql
  username: root
  password: snorbyPASSWD
  host: 10.0.2.23
```

Finálny krok k úspešnemu, plnohodnotnému nasadeniu do prevádzky je nutné ešte nastaviť webový server Apache2, ktorý vyžaduje inštaláciu modulu *PASSENGER* a upresnenie konfigurácie *snorby_config.yml*.

```
development:
  domain: 10.0.2.23
  wkhtmltopdf: /usr/bin/wkhtmltopdf

test:
  domain: 10.0.2.23
  wkhtmltopdf: /usr/bin/wkhtmltopdf
```

```
production:
  domain: 10.0.2.23
  wkhtmltopdf: /usr/bin/wkhtmltopdf
```

5.4.7 Snort

Tento open-source systém Snort v režime IPS je potrebné nainštalovať až po pridaní všetkých potrebných balíčkoch, aby bol zabezpečený správny chod. Stiahnutie zdrojových súborov v najnovšej verzii snort-2.9.2.2 je realizované z domovských stránok projektu [2]. Následne sa súbory z balíčka extrahujú.

```
wget http://www.snort.org/dl/snort-current/snort-2.9.2.2.tar.gz
sudo tar zxvf snort-2.9.2.2.tar.gz
cd snort-2.9.2.2/
sudo ./configure
sudo make && make install
```

Pred konfiguráciou Snortu je ho dobré doplniť o aktuálne pravidlá, podľa ktorých vyhodnocuje, či je prijatá komunikácia v poriadku, či naopak je nevyhovujúca. Balík s aktuálnymi pravidlami verzie snortrules-snapshot-2912 sa sťahuje z domovských stránok projektu [snort]. Avšak tento balík je dostupný pre nekomerčné účely až po registrácií, ktorá nezaberie veľa času. Balík s aktuálnymi pravidlami stiahneme cez webový prehliadač ak sme sa úspešne prihlásili a následne rozbalíme pravidla. Vytvoria sa aj potrebné adresáre pre uloženie záznamu výstrah a pre konfiguračne súbory, do ktorých sa nakopíruje obsah spolu s pravidlami.

```
sudo tar zxvf snortrules-snapshot-2912.tar.gz
cd snortrules-snapshot-2912/
sudo ./configure
sudo make && make install
sudo mkdir /var/log/snort
sudo mkdir /var/local/snort
```

5.4.8 Konfigurácia

Beh systému Snort je závislý na nastavených parametroch v hlavnom konfiguračnom súbore snort.conf, ktorý možno prispôsobiť podľa potrieb. Základné nastavenie dostatočne vyhovuje, preto ostáva nezmenené. Výnimku tvoria premenné definujúce vnútornú sieť a umiestnenie pravidiel. Ako domáca sieť bola zvolená sieť DMZ a ako externá sieť bola nastavená na celú sieť okrem siete DMZ. Vďaka tomuto nastaveniu je možné detekovať taktiež útoky prichádzajúce od užívateľov z vnútornej siete.

```
var HOME NET 10.0.2.0/24
var EXTERNAL NET !$HOME NET
var RULE PATH /etc/snort/rules
```

5.4.9 Spustenie

Posledné kroky vedúce k tomu, aby Snort pracoval v móde inline je, aby modul jádra ip_queue bol spustený. Tento modul zaručuje predávanie paketov do fronty, od kiaľ sú následne vyzdvihnuté a pripravené k ďalšiemu spracovaniu 2.5. Spustenie a kontrola behu tohto modulu jadra je nasledovná:

```
modprobe ip_queue
lsmod | grep ip_queue
```

Aby sa zabezpečilo spustenie modulu jadra spolu so štartom systému, umiestnil som príkaz do shellovského skríptu popísaného v kapitole [44].

```
# Naciatanie modulu ip_queue

/sbin/modprobe ip_queue
```

Záverečný krok je zabezpečenie spustenia Snortu s cestou ku konfiguračnému súboru a adresáru pre zaznamenávanie výstrah.

```
snort -Q -v -c /etc/snort/snort.conf -l /var/log/snort
```

Pričom význam jednotlivých parametrov je:

-Q - zaraďovanie paketov do fronty

-v - výpis na konzolu

-l - cesta k adresáru pre záznam výstrah

-c - cesta ku konfiguračnému súboru

Treba poznamenať, aby sa Snort spustil vždy po štarte systému a najlepšie spoločne spolu s iptables, tak som na koniec skríptu pridal riadok [44].

```
############### Spustenie Snortu #################

/usr/local/bin/snort -Q -v -c /etc/snort/snort.conf -l /var/log/snort
```

6 Testovanie navrhnutého riešenia

Záverečná časť diplomovej práce sa venuje testovaniu navrhnutého riešenia prevencie proti prieniku pomocou penetračných nástrojov. Zameriam sa na základné typy útokov, ktoré je možné detekovať, avšak veľkou mierou je navrhnuté riešenie závislé na použitých pravidlách. Hlavnými kritériami sú typy kontrolovaných protokolov a útokov.

Jednotlivé pravidlá majú obecnú platnosť, ale v niektorých konkrétnych prípadoch je nutné prispôsobenie pre konkrétnu implementáciu. Oficiálne pravidlá, ktoré sú popísané v kapitole [5.4.7], nemusia obsahovať všetky potrebné kontroly, vtedy je nutné vytvoriť si vlastné pravidlá.

6.1 Nmap

Nmap (angl. *Network Mapper*) je open-source nástroj na skúmanie siete a kontrolu bezpečnosti. Zameraný je na rýchle skenovanie veľkých sieti, avšak dobre funguje aj pri nasadení proti jednotlivým hostiteľom. Nmap neobvyklým spôsobom používa neupravené IP pakety na určenie hostiteľských staníc v sieti, služieb, ktoré tieto hostiteľské stanice ponúkajú, na akom operačnom systéme bežia, aký typ paketových filtrov alebo firewallu je použitý, a mnoho ďalších charakteristických vecí. Je ho možne použiť aj pri obvyklých úlohach ako napríklad obsah siete, monitorovanie hostiteľskej stanice alebo služby na nej bežiacej.

Výstup aplikácie Nmap je tvorený zoznamom skenovaných cieľových staníc s dodatočnými informáciami o každom z nich v závislosti na použitých možnostiach. Zaujímavá je tabuľka zaujímavých portov, ktorá obsahuje zoznam čísla portu a protokolu, názov služby a stav. Stav môže byť:

- otvorený (angl. *open*) - aplikácia na cieľovom počítači načúva spojeniam alebo paketom na tomto porte,
- filtrovaný (angl. *filtered*) - nejaký firewall, filter alebo sieťová prekážka blokuje port,
- zatvorený (angl. *closed*) - neobsahujú žiadne na nich načúvajúce aplikácie,
- nefiltrovaný (angl. *unfiltered*) - port reaguje na testovanie, ale nedokážu určiť, či je port otvorený alebo zatvorený

Tento nástroj Nmap sa jednoducho inštaluje z terminálu.

```
sudo apt-get install nmap
```

6.2 Skenovanie portov

V tejto časti sa zameriam na skenovanie portov. Je to postup, ktorým sa snažím získať informácie o poskytovaných službách alebo podporovaných protokolov prípadne taktiež dokáže odhaliť slabinu. Ja sa zameriam, aby moje riešenie prevencie proti prieniku dokázalo úspešne detekovať a následne zahodiť paket, pri niektorých základných typoch skenovania pomocou nástroja Nmap.

Scan TCP SYN Tato metóda skenovania sa odkazuje na polootvorené skenovanie, pretože nie je potrebné otvárať plné TCP spojenie. Odošle sa SYN paket ako keby sa nadviazalo skutočné spojenie a potom sa čaká na spojenie. Prichádzajúci SYN/ACK paket indikuje, že port je otvorený, kým RST, reset paket naznačuje, že port nenačúva. Ak sa neodošle žiadna odpoveď ani po niekoľkých retranzmisiach, port sa označí ako filtrovaný, taktiež aj pri prijatí chybovej správy *ICMP unreachable error*.

Skenovanie portu hostiteľa pomocou nástroja Nmap je nasledovné:

```
nmap -sS 192.168.0.1
```

Ak chcem detekovať v mojom navrhnutom riešení toto skenovanie, je potrebné pridať pravidlo.

```
drop tcp $EXTERNAL_NET any -> any any (msg:"SCAN␣SYN␣-sS"; \
  fragbits: !D; dsize: 0; flow:stateless; flags:S,12; \
  detection_filter: track by_src ,count 10, seconds 60; \
  classtype:network-scan; sid:2000100; rev:2;)
```

Výstup výstravy pri komunikácií a skenovaní portu hostiteľa nástrojom Snort.

```
[**] [1:2000100:2] SCAN SYN -sS [**]
[Classification: Detection of a Network Scan] [Priority: 3]
05/12-23:18:46.901831 192.168.0.2:43221 -> 192.168.0.1:1898
TCP TTL:59 TOS:0x0 ID:2028 IpLen:20 DgmLen:44
******S* Seq: 0x5DC53C2 Ack: 0x0 Win: 0x1000 TcpLen: 24
TCP Options (1) => MSS: 1460
```

TCP Null, FIN a Xmas scany Tieto tri typy skenov využívajú malú dieru v protokole TCP, aby rozlíšili medzi otvorenými a zatvorenými portmi. Ľubovoľný paket neobsahujúci nastavené bity SYN, RST ani ACK vyvolá odpoveď s RST bitom, ak bol zatvorený port a žiadnu odpoveď, ak je port zatvorený. Port je značený ako filtrovaný, ak dorazí chybová správa ICMP unreachable.

Výhodou týchto skenov je, že sa dokažu pretlačiť cez bezstavové firewally a smerovače s filtrovaním paketov. Preto sa pokúsim moje riešené zapojenie prevencie proti útoku otestovať voči takémto typom skenovania a zachytiť výstrahu.

Null scan - toto skenovanie nenastavuje žiadné bity, tcp flag hlavička je nastavená na0.

```
nmap -sN 192.168.0.1
```

FIN scan - nastaví sa bit TCP FIN.

```
nmap -sF 192.168.0.1
```

Xmas scan - nastavia sa flagy FIN, PSH a URG.

```
nmap -sX 192.168.0.1
```

Ak nechceme, aby naša sieť, firewall bol tymito typmi skenovaný, je potrebné vytvoriť pravidlá.

```
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "SCAN␣nmap␣XMAS␣-sX"; \
  fragbits: !M; dsize: 0; flags: FPU,12; ack: 0; window: 2048; \
  reference:arachnids,162; classtype: attempted-recon; sid: 2000111; \
  rev:3; )
```

IPS Snort zachytí tento typ skenu s nastavenými flagmi FIN, PSH a URG a paket zahodí a vygeneruje výstrahu.

```
[**] [1:2000111:2] SCAN nmap XMAS -sX [**]
[Cllasification: Attempted Information Leak] [Priority: 3]
04/30-23:23:13.408003 0:10:F5:A1:26:E2 -> 0:CB:F2:4F:B4:4A type:0x800 len:0x36
192.168.0.2:55661 -> 192.168.0.1:23 TCP TTL:55 TOS:0x0 ID:12511 IpLen:20DgmLen:40
**U*P**F Seq: 0x68134FB6 Ack: 0x0 Win: 0x1000 TcpLen: 20 UrgPtr: 0x0
[Xref => http://www.whitehats.com/info/IDS30]
```

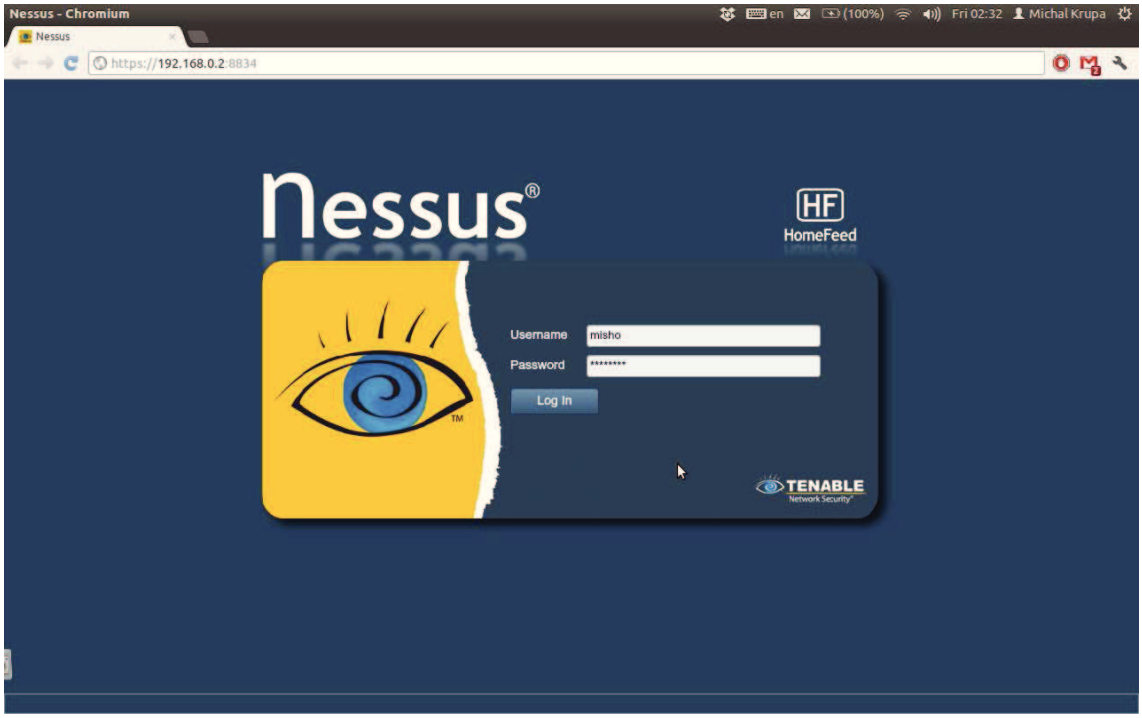
Sken Fin paketu je realizovaný obdobne, pričom sa použije parameter `-sF` a pre skenovanie nulového paketu sa nastavý parameter na `-s0`.

Scan IP protokol Skenovanie protokolu IP určuje, ktoré protokoly IP sú podporované cieľovými počítačmi. Nmap [20] vysiela hlavičky IP paketu a iteruje cez 8 bitové políčko IP protokolu. Hlavičky sú zvyčajne prázdne, neobsahujú dáta a ani patričnú hlavičku pre vyhlásený protokol.

```
nmap -sO 192.168.1.1
```

```
drop ip $EXTERNAL_NET any -> any any (msg:"Protocol␣Scan␣-s0"; \
  dsize: 0; flow:stateless; ip_proto: !icmp; ip_proto: !udp; \
  detection_filter: track by_src ,count 1, seconds 360; \
  classtype:network-scan; sid:2000101; rev:2;)
```

```
[**] [1:2000101:2] Protocol Scan -s0 [**]
[Classification: Detection of a Network Scan] [Priority: 3]
04/30-23:35:53.345390 192.168.0.2 -> 192.168.0.1
PUP TTL:48 TOS:0x0 ID:57213 IpLen:20 DgmLen:20
```



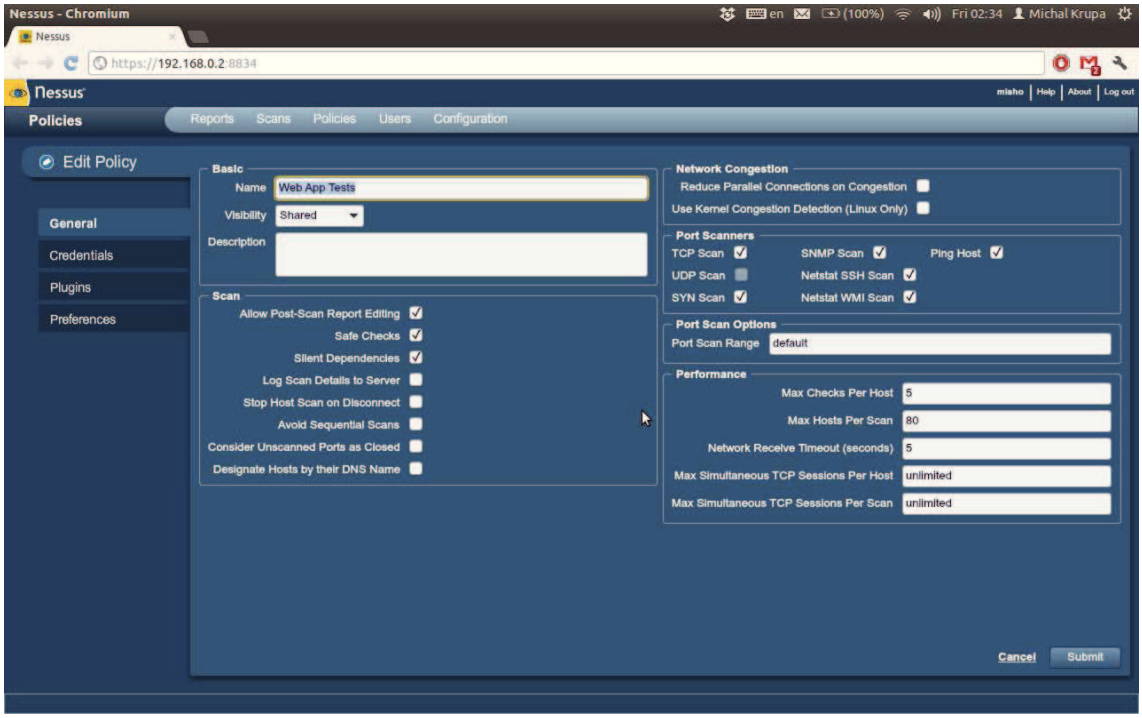
Obr. 13: Úvodná obrazovka Nessus

6.3 Nessus

Je nástroj pre skenovanie vzdialeného počítača a nájdení potenciálnych bezpečnostných dier. Pre osobné použitie je zadarmo. Jeden z najpopulárnejších nástrojov pre skenovanie sieťovej prevádzky vôbec. Obsahuje nespočetné množstvo modulov, ktorými je možný sken. Na server, ktorý vykonáva skenovanie siete, sa pripája klient pomocou webového rozhrania, takže je možné voliť rôzne druhy sieťového skenu z inej siete, akú chceme skenovať.

Základné kategórie, ktoré Nessus dokáže testovať sú:

- získanie shellu
- zraniteľnosť CGI
- vzdialený prístup k súborom
- DOS (angl. *Denial of Service*) útoky
- Firewally
- skenovanie portov
- FTP



Obr. 14: Tvorba pravidies v Nessus

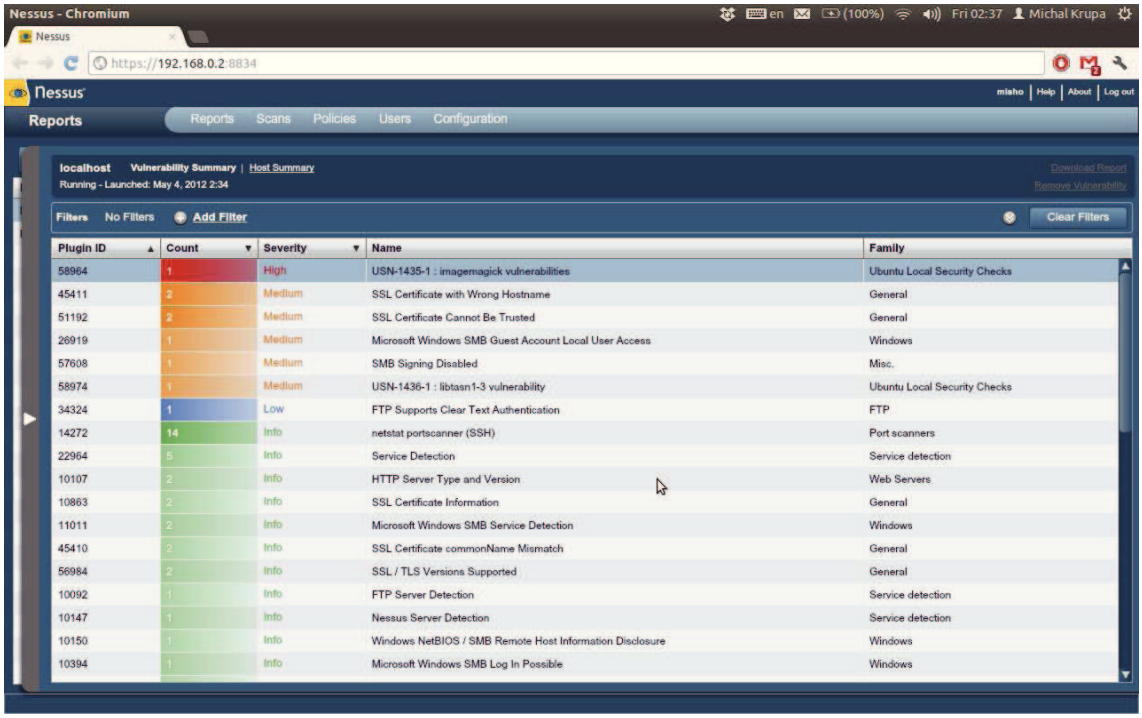
Po stiahnutí balíka z domovských stránok sa nainštaluje.

```
dpkg -i Nessus-5.0.1-ubuntu1110_i386.deb
```

Na server sa pristupuje cez webový prehliadač a to na adrese `https://[server IP]:8834/`, pričom po prvom spustení sa vyžaduje registrácia.

Ako voľbu základnej politiky som zvolil *Web App Tests*, ktorá skenuje zraniteľnosť webovej aplikácie vrátane XSS, SQL a injection. Ako cieľový počítač, ktorý som sa rozhodol skenovať bol môj web server v DMZ zóne, pričom Snort vygeneroval niekoľko výstrah [21].

```
[**] [1:1242:10] WEB-IIS ISAPI .ida access [**]
[Classification: access to a potentially vulnerable web application] [Priority: 2]
04/28-12:01:23.897561 192.168.0.2:39708 -> 192.168.0.1:80
TCP TTL:64 TOS:0x0 ID:39023 IpLen:20 DgmLen:313 DF
***AP*** Seq: 0xD25A9747 Ack: 0xFCC8E7EF Win: 0x16D0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0071][Xref =>
http://www.securityfocus.com/bid/1065][Xref =>
http://www.whitehats.com/info/IDS552]
```



Obr. 15: Výstup skenovania siete v Nessus

6.4 Vlastné pravidlá

Niekedy je vhodné, navzdory už existujúcim pravidlám vytvoriť si vlastné pravidlo, ktoré by vyhovovalo aktuálnym požiadavkám. Takéto pravidlo, v mojom prípade `mojepravidla.rules` sa potom pridá do adresára so stiahnutými pravidlami */etc/snort inline/rules* a jeho cesta sa pridá do konfiguračného súboru Snortu *snort.conf*.

```
include $RULE_PATH/mojepravidla.rules
```

Napríklad pravidlo na zakázanie prístupu na webové stránky www.ulozto.cz, ktoré kontroluje TCP pakety z vonkajšej siete. Zisťuje, či paket obsahuje doménovú adresu. V prípade zhody je zahodený.

```
drop tcp $EXTERNAL_NET any -> any any \
(msg:"DROPPED Connecting to www.ulozto.cz"; content:"www.ulozto.cz"; \
  nocase; classtype: web-application-activity;sid:1000007; rev:1;)
```

Snort vytvorí výstrahu a prevedie zahodenie.

```
[**] [1:1000007:1] DROPPED Connecting to www.ulozto.cz [**]
[Classification: access to web application] [Priority: 2]
04/28-13:43:33.600724 10.1.1.100:1101 -> 93.99.92.193:80
```

```
TCP TTL:127 TOS:0x0 ID:365 IpLen:20 DgmLen:546 DF
***AP*** Seq: 0xF6255B0C Ack: 0x41376A9B Win: 0xFFFF TcpLen: 20
```

Súbor `mojepravidla.rules` s týmto a inými pravidlami je priložený na záznamovom médiu, ktoré je súčasťou diplomovej práce [22].

7 Záver

Ja osobne chcem poukázať na to, že v dnešnej spoločnosti sa nekladú také veľké nároky na bezpečnosť. Bezpečnostná stratégia obsahuje iba základné ochranné prvky ako sú napríklad firewally alebo antivírusy. Ich nasadenie v praxy je limitované svojou funkcionalitou na veľmi úzky okruh pôsobnosti. Ak je potrebné pokryť širšie spektrum možných hrozieb, využívajú sa ďalšie prvky ochrany.

Systém prevencie proti narušeniu je práve jeden zo systémov, ktoré dokážu na základe svojích pravidiel kontrolovať široký rozsah pôsobnosti a tak brániť neoprávneným vstupom. Výhodou je, že je možné ich kombinovať s inými prvkami pre zabezpečenie siete. Realizované môžu byť ako samostatný hardware alebo ako softvérové riešenie. Sieťová prevádzka, ktorá prejde cez firewall je ukladaná do fronty, z ktorej si ju systém prevencie proti prienikom vyzdvihne a aplikuje na ňu vlastné pravidlá. Súčasťou je taktiež príslušná akcia, ktorá sa ma vykonať v prípade nájdenia nežiaduceho obsahu či udalosti.

V mojej navrhnutej sieti boli použité prvky ako firewall a systém na prevenciu proti prienikom. Kombinácia týchto dvoch prvkov prináša vyššiu mieru zabezpečenia. Kým firewall založený na iptables so stavovou paketovou filtráciou má nastavené pravidla udávajúce, aká sieťová prevádzka je schopná cez neho prejsť, prípadne aka má byť zablokovaná alebo odmietnutá. Tak systém prevencie proti prienikom Snort, umiestnený na firewalle, je schopný detekovať závažnú komunikáciu smerujúcu predovšetkým do vnútornej siete. Výstrahy generované systémom prevencie proti narušeniu, ktoré vznikajú pri závažnej komunikácií, sú zasielané do databázy. Vďaka tejto možnosti je možné robiť štatistické operácie a mať tak pod dohľadom svoju sieť z pohodlia svojho domova prístupom cez webové rozhranie. V prípade neštandardného chovania na sieti je možné kedykoľvek pridať svoje vlastné pravidlo a tak eliminovať anomálie.

Navrhnuté riešenie bolo testované vybranými útokmi pomocou penetračných nástrojov, ktoré mali za úlohu získať informácie o vnútornej sieti a prevádzkovaných protokolov. Dosiahnuté výsledky poukazujú na to, že aj pri použití dvoch bezpečnostných prvkov je potrebné sieťovú prevádzku neustále monitorovať a pridávať nové pravidlá, aby ta nežiaduca komunikácia bola odhalená. Tu nastáva problém, čím robustnejšie bezpečnostné systémy, tým je nutné dodávať dostatočné množstvo kvalitných pravidiel.

Michal Krupa

8 Literatúra

[1] Endorf, C., Schultz, E., Mellander, J *Intrusion Detection Systems with Snort,* Pearson Education 2003.

[2] *Snort* [online]. Dostupný z WWW: <http://www.snort.org>

[3] *OSI model* [online]. Dostupný z WWW: <http://en.wikipedia.org/wiki/OSI_model>

[4] *OSI model* [online]. Dostupný z WWW: <http://www.ietf.org/rfc/rfc1519.txt>

[5] *Ipchains* [online]. Dostupný z WWW: <http://www.cs.vsb.cz/grygarek/TPS-0304/projekty0304/ipchains/3/Firewall.pdf>

[6] *Firewall obrnte sve pocítace* [online]. Dostupný z WWW: <http://pctuning.tyden.cz/software/ochrana-pocitace/4296-firewall-obrnte_sve_pocitace>

[7] *Firewall* [online]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Firewall>

[8] *cesnet* [online]. Dostupný z WWW: <http://www.cesnet.cz/akce/2009/bezpecnost-siti/p/bezpecnost-fw.pdf>

[9] *Linuxový Firewall - základy iptables* [online]. Dostupný z WWW: <http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-linuxovy-firewall-zaklady-iptables>

[10] *SSL on Apache2 on Ubuntu* [online]. Dostupný z WWW: <http://beginlinux.com/blog/2009/01/ssl-on-ubuntu-810-apache2/>

[11] *Certificate Authority* [online]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Certificate_authority>

[12] *Create a FTP server with user access (proftpd)* [online]. Dostupný z WWW: <http://ubuntuforums.org/showthread.php?t=79588>

[13] *Pcap* [online]. Dostupný z WWW: <http://sk.wikipedia.org/wiki/Pcap>

[14] *tcpdump* [online]. Dostupný z WWW: <http://www.tcpdump.org/>

[15] *libdnet* [online]. Dostupný z WWW: <http://libdnet.sourceforge.net/>

[16] *libnet* [online]. Dostupný z WWW: <http://ips-builder.googlecode.com/>

[17] *pcre* [online]. Dostupný z WWW: <http://www.pcre.org>

[18] *Barnyard2* [online]. Dostupný z WWW: <http://www.securixlive.com/barnyard2/>

[19] *Snorby* [online]. Dostupný z WWW: <http://snorby.org/>

[20] *Nmap* [online]. Dostupný z WWW: <http://nmap.org/man/>

[21] *nessus* [online]. Dostupný z WWW: <http://www.tenable.com/products/nessus>

[22] *Working with Snort Rules* [online]. Dostupný z WWW: <http://www.pearsonhighered.com/samplechapter/0131407333.pdf>

A Najjednoduchší, nestavový firewall

Teraz na základe získaných dovedností, ktoré som predstavil doposiaľ predstavím najjednoduchší firewall, ktorý ide zostaviť. V prvom rade je potrebné vyčistiť všetky pravidlá.

```
iptables -F
```

Keď sú všetky pravidlá vyčistené, špecifikujú sa jednotlivé pravidlá, ktoré zabezpečia povolenie všetkej prevádzky z miestneho rozsahu 10.0.0.0/8, prístup k SSH (skratka) iba z adresy 11.22.33.44 a povolenie prístupu k web serveru a k SMTP serveru.

```
iptables -A INPUT -s 10.0.0.0/8 -j ACCEPT
iptables -A INPUT -s 11.22.33.44 -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 25 -j ACCEPT
```

Je všeobecne známe, že pri tvorbe firewallu sa používa stratégia, že sa všetko zakáže a povolí sa len to potrebné, preto pri špecifikácií východziej politiky sa nastaví u reťaze INPUT politiku DROP.

```
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
```

B Hostiteľský alebo sieťovo orientovaný systém detekcie narušenia

NIDS	**HIDS**
široké použitie (kontroluje všetký sieťové činnosti)	úzke použitie (kontroluje iba špecifickú činnosť hostiteľa)
jednoduchšie nastavenie	zložitejšie nastavenie
vhodnejšie pre detekciu vonkajšieho napadnutia	vhodnejšie pre detekciu vnútorneho napadnutia
lacnejšia implementácia	drahšia implementácia
detekcia je zaznamenaná z celej siete	detekcia je zaznamenaná z jednotlivých hostiteľov
preskúmavá hlavičku paketu	ignoruje hlavičku paketu
odozva takmer v reálnom čase	odozva až po podozrivom vstupe v logovacom súbore
nezávislý na operačnom systéme	operačný systém musí byť špecifický
sieťový útok detekuje ako dôsledok analýzy užitočnej záťaže	detekuje lokálny útok skôr ako je napadnutá vlastná sieť
detekuje neuspešné pokusy o útok	verifikuje úspech alebo zlihanie útoku

Tabuľka 1: Hostiteľský alebo sieťovo orientovaný systém detekcie narušenia

C IDS verzus IPS

IDS	**IPS**
Inštalujú sa na segment siete (NIDS) a uzol (HIDS).	Inštalujú sa na segment siete (NIPS) a uzol (HIPS).
V sieti sú pasivným prvkom.	Sú radené sériovo (nie sú pasívne).
Nemôžu analyzovať šifrovanú prevádzku.	Vhodnejšie pre ochranné aplikácie.
Centrálna správa riadenia.	Centrálna správa riadenia.
Vhodnejšie pre detekciu útokov.	Ideálny pre blokaciu webovských znetvorenín.
Výstrahu vydavajúci produkt (reaktívny).	Blokujúci produkt (preaktívny).

Tabuľka 2: IDS verzus IPS